A CONTROLLED SENSING APPROACH TO GRAPH CLASSIFICATION

BY

JONATHAN GILLMORE LIGO

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2013

Urbana, Illinois

Adviser:

Professor Venugopal Varadachari Veeravalli

ABSTRACT

Graphs are used to model dependency structures, such as communication networks, social networks, and biological networks. Observing the graph in its entirety may be undesirable due to size of the graph or noise in observations, especially if only a function of the graph structure is of interest, such identifying one of finitely many classes to which the graph belongs. In this thesis, we develop a framework for jointly classifying a graph and sampling a graph in order to maximize the decay of classification error probability with sample size by formulating the classification problem as a composite sequential hypothesis test with control. In contrast to prior work, posing the problem as a composite sequential hypothesis test with control provides provable performance guarantees through the controlled sensing framework and allows the classification problem to improve the quality of observations in the sampling procedure.

The algorithm proposed in this thesis is demonstrated by classifying graphs with respect to average node degree as a measure of connectivity. Observations of the graph are collected by selecting a node to sample and observing some subset of possible edges in the complete graph incident to the node according to two probability models, where observations are conditionally independent given their neighborhoods in the graph. Simulations are provided for an Erdös-Rènyi graph to show the trade-off between sample size and classification performance and show that the proposed algorithm outperforms a random walk-based technique.

To my family, for their love and support

ACKNOWLEDGMENTS

First, I would like to thank my adviser, Professor Venugopal Veeravalli for advising me through this work. The primary contribution of this thesis (in Chapter 4) was joint work with Professor George K. Atia at the University of Central Florida resulting in our 2013 publication, "A Controlled Sensing Approach to Graph Classification" (see reference list). I would also like to thank my friends, research groupmates, officemates, and graduate students of the communications group in the Coordinated Science Laboratory for their support. Finally, I would like to thank the United States Defense Threat Reduction Agency for funding this work through subcontract 147755 at the University of Illinois from prime award HDTRA1-10-1-0086 as well as the University of Illinois at Urbana-Champaign's ECE Distinguished Fellowship and the Joan and Lalit Bahl Fellowships.

TABLE OF CONTENTS

CHAPTER 1

AN INTRODUCTION TO GRAPHS AND THEIR APPLICATIONS

Many datasets of interest in engineering and the physical and social sciences consist of data subject to constraints on dependencies, giving rise to a network structure. The natural framework for modeling these dependencies is through a graph. We first highlight some applications. In biology and bioinformatics, graphs are used to study the relationships in how genes are expressed by performing clustering on the gene network [1]. In the physical sciences, random graphs are often used in the context of statistical mechanics, such as in the study of the Ising model. The insights from connections in statistical mechanics have been adapted to study large networks of interest to engineers and social scientists, such as social networks (e.g., Facebook) [2]. Graphs also represent models of communication structures, such as the internet and telecommunication networks [3, 4]. A graph representing the IPv4 and IPv6 internet is shown in Fig. 1.1.

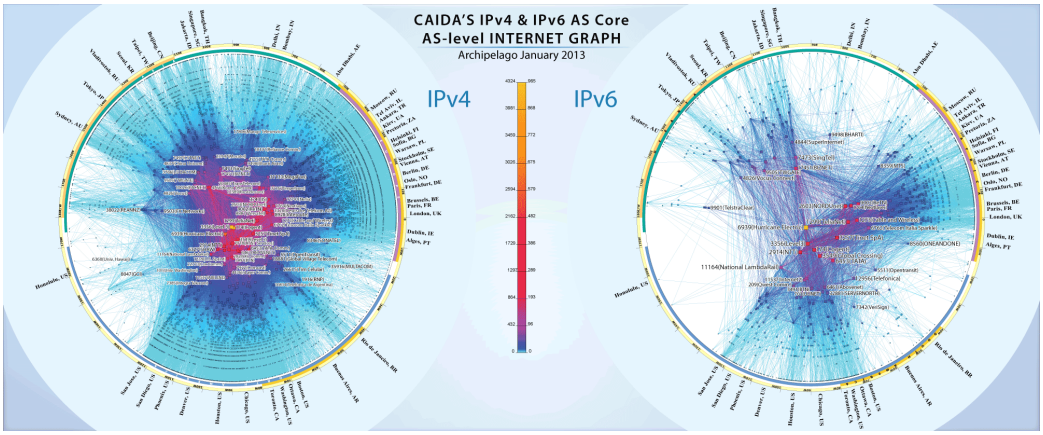


Figure 1.1: A graph of the structure of the internet as of January 2013 [5].

There are two major problems of interest about graphs: graph inference and graph sampling. The task of graph inference is to draw a conclusion based on a graph sample, such as in the case of gene expression. The

task of graph sampling is to efficiently acquire a representative sample of an (often large) graph, as in the case of studying online social networks [6, 7] or observing the structure of the internet as in Fig. 1.1. Sampling is necessary as the size of graphs arising in practice may be too large for useful inference (for example, the Facebook social network currently has over one billion users) or may be corrupted by noise (missing or spurious edges or nodes). In this thesis, we present a joint formulation of these problems for inferring measures of connectivity in the presence of noise.

Determining if the connectivity of a graph is high or low gives some insight into the operating point of the system modeled by the graph. In [8], disease spreading in a population is modelled as a process on the population graph, with connectivity measured through shortest paths between people. In this case, connectivity serves as a measure of how quickly a population can be infected. Classifying this graph as highly connected serves as a warning of potential epidemics. Measuring connectivity will be discussed later in Section 1.3, and a formulation of the inference and sampling problem to classify a graph as highly connected or not highly connected will be described in Chapter 4.

1.1 Graph Theory Preliminaries

Most of the terminology presented is consistent with (or a simplified version of) that used in graph theory textbooks, such as [9]. We refer the reader to [9] for more details.

A *graph* (or *network*) is a pair of sets (V, E) where V is the set of *vertices* (also called *nodes*) and *edges* $E \subset V \times V$. We will assume $1 \leq |V| < \infty$ but we will allow $|E| = 0$ (though this case is typically not of interest). If we consider the edge $e_{ij} \triangleq (i, j) \in E$ distinct from the edge e_{ji}, we say the graph is *directed* (and there are $|V|^2$ possible edges). If we consider the edge e_{ij} identical to the edge e_{ji}, we will disallow *loops* (that is, $e_{ii} \notin E$ for all $i \in V$) and say the graph is *undirected* (and there are $\binom{|V|}{2}$ possible edges). The *endpoints* of edge e_{ij} are the vertices i and j. Unless otherwise stated, a graph is considered to be undirected (in graph theory literature, the graphs we will consider are often called *simple graphs*). The *complement* of a graph G, denoted $G^C = (V, E^C)$, is the graph on the same vertices which contains

all possible edges on vertex set V other than those in E (that is, E^C is the complement of the edge set relative to the set of all possible edges). If vertices i and j have an edge between them in E, we say they are *neighbors* (or *adjacent*). The edge e_{ij} is then said to be *incident* to vertices i and j. The set of all neighbors of a vertex i in graph G is its *neighborhood* and is denoted $N_G(i)$. The degree of vertex i, d_i, is its number of neighbors. Often, it is natural to take $V = [N] \triangleq \{1, 2, \ldots, N\}$ for some positive integer N (and thus, we will often use i and j to refer to vertices of graphs). Note for every directed graph $G_d = (V_d, E_d)$, we can associate an undirected graph $G = (V, E)$ called the *underlying graph* where $e_{ij} \in E$ if and only if $e_{ij} \in E_d$ or $e_{ji} \in E_d$ (that is, we ignore the directions imposed by the edges in G_d). In some cases, the underlying graph of a directed graph can capture much of the interesting structure of the directed graph.

A *subgraph* of a graph $G = (V, E)$ is a graph $G' = (V', E')$ such that $V' \subset V$ and $E' \subset E$ subject to the constraint that for all $e_{ij} \in E'$, $i, j \in V'$. For notational convenience, this is denoted $G' \subset G$. A *path* from vertex i to j is a subgraph whose edges form a sequence $e_{ii_1}, e_{i_1i_2}, \ldots, e_{i_mj}$. The smallest number of edges in a path between distinct vertices i and j (∞ if no such path exists) is the *distance* between i and j. A graph is *connected* if there exists a path between any pair of distinct vertices (i.e., the distance between any pair of distinct vertices is finite).[1] A maximal connected subgraph is called a *component* of the graph. Note that a graph is connected if and only if it has one component.

To clarify the difference between a directed and undirected graph and demonstrate the former properties, one can consider a road network, which can be represented as a directed graph. The vertices consist of exits on the roads, and edge e_{ij} exists in the graph if there is a road whose traffic goes from exit i to exit j. Note that edge e_{ij} is different from edge e_{ji} as there may exist a road segment going from i to j but not vice versa, so the directed nature of the graph is inherent in the underlying structure (for example, driving while ignoring this structure may cause a head-on collision and is thus not recommended). The road network within a contiguous landmass is often designed to be connected to allow users to reach any point of interest on the landmass.

[1] We will discuss connectivity in Section 1.3, which is a different notion from a graph being connected, and our definition will contain the usual graph-theoretic meanings.

A simple example of an undirected graph is a graph of friendships where the vertices represent persons and edge e_{ij} exists if and only if person i is friends with person j. Note since person i is friends with person j if and only if person j is friends with person i, friendship should be a symmetric relationship. A social network such as Facebook does not necessarily obey the symmetry constraints, because relationships can be made uni-directional (such as "Liking" the page of a band represents a connection from a fan to the band, but not vice versa). In order to account for the uni-directionality, a directed graph is required. However, the underlying graph captures most of the interesting structure. Depending on the number of people considered, the graph of friendships may be disconnected (such as the jocks in a high school class not sharing friends with the marching band).

In fact, when the underlying graph of many social phenomena is considered, it is often effectively connected due to its size. This result has been observed in many cases such as *Milgram's small world experiment*, where persons were instructed to forward a package to their friends in order to reach some given individual. While most of the packages were lost, those which reached their destination took about six people to reach their destination, leading to the phrase "six degrees of separation" [2]. Several other datasets exhibit similar features, such the graph of collaborations between mathematicians. The Erdös number of a mathematician measures the minimum distance between the mathematician and Paul Erdös in the collaboration graph. A numerical study of about $401,000$ mathematicians showed most mathematicians have an Erdös number less than 9 [10]. Similarly, the Erdös-Bacon number [11] is used to measure distances in the collaboration graph of mathematicians (by distance to Erdös) and actors (by distance to Kevin Bacon).

In order to work with graphs mathematically, we will need a concise representation of a graph. We highlight four such ways which will be useful in this thesis:

1. *Adjacency List*: An adjacency list consists of a list of the edges in the graph (along with a separate listing of the vertex set). This graph representation requires $|V| + 2|E|$ units of storage. This is useful for representing *sparse* graphs – that is, ones with not many edges (e.g., $O(|V|)$ edges where $O(\cdot)$ is given as big-O notation). While efficient for storage and implementation, adjacency lists are unwieldy to math-

ematically manipulate compared with the following representations.

2. *Adjacency Matrix*: An adjacency matrix is a $|V| \times |V|$ matrix A where $A_{ij} = 1$ if and only if $e_{ij} \in E$. Note this matrix is symmetric for an undirected graph, and possibly asymmetric for a directed graph. Note that the row sums give the degrees of each vertex.

3. *Laplacian Matrix*: The Laplacian matrix L is given by $L = \text{diag}(d_1, \ldots, d_{|V|}) - A$ where $\text{diag}(a_1, \ldots, a_n)$ denotes the $n \times n$ diagonal matrix with $a_1, \ldots, a_n$ on the diagonal. The eigenvalues and eigenvectors of this matrix form the basis of *spectral graph theory*.

4. *Incidence Matrix*: The incidence matrix is a $|V| \times |E|$ matrix where the column corresponding to edge e_{ij} has ones in indices i, j and zeroes elsewhere.

In this thesis, we will primarily focus on using the adjacency matrix and incidence matrix representations of the graphs. Computer implementations of graph algorithms use adjacency lists for storage efficiency. We will briefly mention connectivity measures derived from the Laplacian matrix in Section 1.3.

1.2 Sparsity and Random Graphs

In many applications, graphs are *sparse* – that is, each vertex is connected to relatively few other vertices. The precise mathematical formulation of sparsity of a graph is dependent on application, but usually involves the size of the edge set to be controlled by some sub-quadratic function of the size of the vertex set, such as $|E| = O(|V|^\alpha)$ where $0 \leq \alpha < 2$. Sparsity is a natural structural property present in many graphs, such as social networks, where most people are not connected to each other.

In analyzing graph algorithms, the use of *random graphs* is often convenient. A random graph is a probabilistic construction of a graph on a given number of vertices (that is, it is a graph-valued random process indexed by number of vertices). Typically, random graphs can be constructed for any number of vertices. Constructions of random graphs can be broadly classified by the graph-theoretic properties captured. In this section, we will

highlight the construction of Erdös-Rènyi random graphs along with Watts and Strogatz random graphs. We will say a property holds for *almost every graph* if the fraction of graphs on n vertices for which the property holds goes to 1 as n goes to infinity.

The most basic random graphs are the Erdös-Rènyi (ER) random graphs. An *Erdös-Rènyi model A* random graph with parameter $p \in [0,1]$ on n vertices consists of including edge e_{ij} in the graph with probability p independent of other edges. Thus, the Erdös-Rènyi model A graph with $p = \frac{1}{2}$ uniformly chooses a graph on n vertices. Unless otherwise noted, an Erdös-Rènyi model A graph has parameter $p = \frac{1}{2}$. An *Erdös-Rènyi model B* graph on n vertices with parameter m has a uniform distribution on all graphs of n vertices with m edges. When n is large and $p = \frac{m}{\binom{n}{2}}$, the ER model A graph approximates the ER model B graph. This is useful as the ER model B graph is harder to work with mathematically than the ER model A graph. Using a concentration inequality, one can see that an ER model A graph concentrates closely to having np edges and if p is not chosen to be a function of n, almost every ER model A graph is connected [9]. Note that it is simple to generate graph samples from the ER model A random graph.

ER model A graphs are favored for use in the *probabilistic method*. The probabilistic method uses probability theory and concentration inequalities to show the existence of graphs with certain properties, or to show a property holds on almost every graph. While the mathematical properties of ER model A graphs are good, numerical validation of algorithms by averaging using ER model A graphs is difficult since there are $2^{\binom{n}{2}}$ graphs on n vertices. Note that this model does not enforce sparsity, which is crucial for accurately modeling large-scale graphs that arise in applications, such as social networking data sets. We will show numerical results using the Erdös-Rènyi model A random graph for the algorithm proposed in Chapter 4.

A more complex random graph which received much attention from the mathematics community was proposed by Watts and Strogatz in [12]. A random graph with the *Watts and Strogatz* model with parameters $p \in [0,1], k < n$ on n vertices is constructed as follows:

1. Place the n vertices on a circle.

2. Connect each vertex to its k nearest neighbors on the circle.

3. Iterating from $i = 1, \ldots, \frac{k}{2}$, walk around the circle, randomly reconnecting the edge between the considered vertex and its i-th neighbor on the circle with probability p.

By tuning p and k, the connectivity properties of the graph can be tuned more easily than the random graph. A downside of this construction is that it is mathematically difficult to work with the Watts-Strogatz model. Several modifications for increased mathematical tractability are discussed in [2], but they do not result in graphs as discussed in this work. Note that p imposes some structure on this random graph – if p is sufficiently low, the graph is approximately regularly connected between the vertices, while if p is sufficiently high, it is close to an Erdös-Rènyi model B graph with $m = \frac{nk}{2}$.

Some work has also been done on generating graphs with particular *degree distributions*, where the degrees of vertices following a given distribution such as a power law distribution (leading to *scale-free networks*) [2]. For an arbitrary degree distribution, the *configuration model* is a random graph with a uniform distribution over all graphs with the prescribed degree distribution. An appropriately constructed configuration model allows the capture of more structure present in real networks, such as the World Wide Web [2], than Erdös-Rènyi graphs.

In simulations, we use Erdös-Rènyi model A graphs due to their simplicity as well as the size of the graphs involved. The added complexity of other random graphs is primarily useful for simulating graphs much larger than those simulated.

1.3 Notions of Connectivity

All notions of connectivity mentioned will be for undirected graphs – some can be extended in a straightforward manner to directed graphs as well.

Graph theory is primarily concerned with two notions of connectivity, known as (vertex) connectivity and edge connectivity. The *(vertex) connectivity* is the minimum number of vertices which need to be removed from the graph in order to disconnect the graph (i.e., make it not connected) or leave one vertex. A graph with vertex connectivity greater than or equal to k is said to be k-connected. A motivating application for vertex connectivity is an ad-hoc file sharing network. Assume a file must be transmitted to

everyone on a network where users are allowed to leave the network. The file can be transmitted to all remaining users on the network after any k users leave if and only if the graph representing the network is k-connected (else, there would be a portion of the network which could not be reached). The *edge connectivity* is the minimum number of edges which need to be removed from the graph to disconnect the graph. The study of edge connectivity can be motivated with a dual of the prior problem, where the links in the network can fail but the file must still reach every user on the network. A graph with edge connectivity greater than or equal to k is said to be k-edge-connected. The file can be transmitted to all users on the network when any k links fail if and only if the graph representing the network is k-edge-connected. While vertex and edge connectivity are useful in applications where the graph studied models a network designed for fault tolerance (such as quantify worst-case failure scenarios for a data network), they do not capture information of interest to applications such as social network analysis. To this end, we introduce a few notions of connectivity which are useful for other applications, such as social network analysis.

A few measures of connectivity can be found via eigendecomposition of the Laplacian matrix studied in spectral graph theory. It is obvious from Gershgorin's circle theorem that the Laplacian is positive semidefinite. The second smallest eigenvalue is known as the *algebraic connectivity*, which is positive if and only if the graph is connected. Among other nice properties, this eigenvalue exhibits *interlacing* – it changes by at most 2 by addition of an edge (similar to the monotonicity property we will discuss later in this section). The spectral graph properties give rise to bounds on various graph theoretic properties such as connectivity. For more details, the reader is referred to Chapter 13 of [13].

We now begin discussion of notions of connectivity which are useful for dealing with datasets collected from graphs. The *mean geodesic path length* of a graph is the arithmetic mean of the shortest path lengths between any distinct pair of vertices. If the network is not connected, one can replace the arithmetic mean with the harmonic mean or only consider pairs of vertices which are connected via a path [2]. This measure of connectivity quantifies the small-world-like nature of some networks and has applications to processes propagating along a network, such as a disease spreading [8] or the aforementioned Erdös numbers and Erdös-Bacon numbers in a social

context. While this measure of connectivity has many applications, there is no closed-form expression for the shortest path length between two vertices in a graph, and slight changes to a graph can drastically change the shortest path length between two vertices (and thus, the mean geodesic path length). Consider the example of the highway network in the continental United States and Hawaii. Clearly, these networks are not connected, so the mean geodesic path length is infinite. However, adding an edge (a road) between the Hawaiian islands and some point in the continental United States results in a finite mean geodesic path length.

The *clustering coefficient* can refer to two related quantities in literature [2]. The first, derived from the sociological side of social networking literature as the *fraction of transitive triples* or *transitivity (coefficient)*, is given by

$$T = \frac{3 \times \text{number of triangles in the graph}}{\text{number of connected triples of vertices}} \tag{1.1}$$

where a *triangle* is a set of three vertices having edges with each other and a *connected triple of vertices* is a set of three vertices which lie on a path of length two. Clearly, the number of connected triples of vertices is half the number of paths of length two, and thus, we can write T as

$$T = \frac{\text{tr}(A^3)}{\sum_{i \neq j} [A^2]_{ij}} \tag{1.2}$$

where A is the adjacency matrix of the graph and $\text{tr}(A)$ is the *trace* of A (the sum of its diagonal entries or, equivalently, the sum of its eigenvalues). Note the transitivity satisfies $0 \leq T \leq 1$, justifying the use of coefficient in the name. Equation (1.2) can be seen by noting $[A^n]_{ij}$ is the number of paths to get from vertex i to vertex j through n edges. So, $\frac{\text{tr}(A^3)}{6}$ is the number of triangles in the graph (since each triangle is counted three times by each vertex it contains, and twice for each vertex by the ordering of the vertices visited in the path) and similarly, $\sum_{i \neq j} [A^2]_{ij}$ is the number of paths of length two. Note that while A is a binary matrix, if one relaxes its entries to lie within the interval $[0, 1]$, it can be shown that T is not convex in the entries of A. Thus, estimating a graph with transitivity constraints cannot be solved as a convex programming problem and makes this quantity difficult to implement in the work of Chapter 4. Also, note that the transitivity coefficient relies on the cube of the adjacency matrix, which

is computationally expensive for large graphs. The functional interpretation of transitivity when the graph is a social network is "the mean probability that the friend of your friend is also your friend" [2].

The second definition of clustering coefficient, C, was proposed by Watts and Strogatz in [12]. The *local clustering coefficient* is given by

$$C_i = \frac{\text{number of triangles containing vertex } i}{\text{number of triples with } i \text{ as the middle vertex}} \tag{1.3}$$

where vertices with 0 or 1 neighbors have $C_i = 0$. The *(global) clustering coefficient* is given by

$$C = \frac{1}{N} \sum_{i \in [N]} C_i \tag{1.4}$$

which gives a higher weight for clustering about low degree nodes versus the transitivity coefficient. It is important to note that for many networks of interest, this quantity is expected to be non-zero as the network size increases. However, for the ER model A graph it (as well as the transitivity coefficient) can be shown to be $O(N^{-1})$ on N vertices. Thus, while the ER model A graph is extremely convenient from a mathematical point of view, it does not capture enough structure for the study of some real-world datasets.

The notion of connectivity we will focus on in this thesis is average node degree. If $\boldsymbol{d}_G$ is the degree vector for graph G with $V = [N]$, then the average node degree $\bar{d}_G$ is given by

$$\bar{d}_G = \frac{1}{N} \boldsymbol{d}_G^\top \mathbf{1}_N \tag{1.5}$$

where $\mathbf{1}_N$ is the $N \times 1$ vector of ones. A motivating example for average node degree to measure connectivity is in the case of a social network, where the average node degree represents the average number of connections per user. If the average number of connections is high (low) with respect to some threshold, then the graph will be considered highly (not so) connected. Note this quantity also has several nice computational aspects – adding (removing) one edge increases (decreases) this quantity by $\frac{2}{N}$, and writing

$$\boldsymbol{d}_G = A_G \mathbf{1}_N \tag{1.6}$$

we see the average node degree is a convex (in fact, linear) function of the

entries of the adjacency matrix. The average node degree is also *monotone* – that is, if $G' \subset G$, $d_{G'} \leq d_G$. We will use this property to reduce the complexity of our proposed algorithm in Chapter 4. It is easy to construct examples for the other notions of connectivity mentioned to show non-monotonicity.

Several other quantities can be thought of as notions of connectivity in a network, such as functions of community size (discussed in Section 1.4). More details on this notion and others can be found in Newman's review paper [2].

1.4 Problems in Graph Analysis

We can broadly divide problems about graphs into the classes of graph sampling and graph inference. These problem classes are not disjoint – in fact, the contribution of this thesis is to formulate a joint sampling and inference scheme for graphs as in Chapter 4.

The problem of *graph sampling* is to design an algorithm to acquire a representative sample of a graph. This is of interest particularly when graphs are large, such as graphs representing social networks, due to both storage constraints as well as the need to reduce computational complexity [6]. Typically, algorithms used for this problem are formulated as random walks on the graph. We discuss this problem in more detail in Chapter 3.

The problem of *graph inference* is to calculate some function of the graph. One common example with applications to fields such as machine learning, communications, and signal processing is the problem of parameter estimation. Assuming the graph was drawn according to some random graph model, one can estimate the parameters of the model from which the graph was drawn (using an approach such as maximum-likelihood) or the nearest graph under that model such as in [14]. Another major problem of interest is graph clustering and the detection of community structure [15]. In this problem, one wants to detect groups of nodes which can be grouped into communities (such as lumping students by cliques in a high school social network or detecting closely related groups on the internet). Note the natural ordering of these problems is to first perform graph sampling then graph inference.

The problem which we consider in this thesis is *graph classification*, which is a subproblem of graph inference. Graph classification refers to deciding if

a graph belongs to one of finitely many classes, such as the classes "highly connected" and "not so connected." By posing this problem as a sequential hypothesis test, a special case of graph inference and sampling are formulated jointly and solved simultaneously in the presence of noise where the sample is missing some edges and has some spurious edges.

1.5 Notation

Here, we summarize some relevant (predominantly graph-related) notation for this thesis.

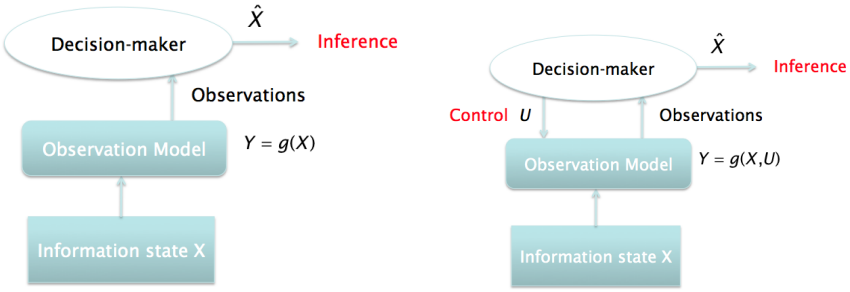
- ***Bold*** face will be used for vectors, with subscripts indicating components. *Italics* will be used for definitions. Matrices will be in plain face.
- $\boldsymbol{x}^n$ is the vector $(x_1, \ldots, x_n)$.
- $f(n)$ is $O(g(n))$ (read f is big-Oh of g) for some non-negative function $g(n)$ if there exists n_0 such that for all $n \geq n_0$, $|f(n)| \leq Cg(n)$ where $C > 0$ is a constant.
- $f(n)$ is $o(g(n))$ (read f is little-Oh of g) for some non-negative function $g(n)$ if for any $C > 0$ there exists $n_0 = n_0(C)$ such that for all $n \geq n_0$, $|f(n)| \leq Cg(n)$.
- $[n] = \{1, \ldots, n\}$.
- Graph $G = (V, E)$ will often have $V = V_G, E = E_G$ to denote which vertex set, edge set corresponds to the graph.
- $\widehat{a}$ denotes an estimate a (such as a maximum likelihood estimate (MLE)).
- A^C denotes the complement of the set A.
- $A = A_G$ denotes the adjacency matrix of a graph G.
- $M = M_G$ denotes the incidence matrix of a graph G.
- $N_G(i)$ denotes the neighborhood of vertex i in graph G.
- $\boldsymbol{d}_G$ denotes the vector of degrees of vertices in graph G.

- $\mathrm{diag}(a_1, \ldots, a_n)$ is the $n \times n$ diagonal matrix with $a_1, \ldots, a_n$ on its diagonal.
- $\mathrm{tr}(A)$ is the trace of the square matrix A.

CHAPTER 2

HYPOTHESIS TESTING AND CONTROLLED SENSING

In this section, we first overview some results from hypothesis testing without control for the finite sample-size case and sequential case. Then, we present a brief introduction to the controlled sensing paradigm. A *hypothesis test* (also called a *decision rule* or *classifier*) is a rule which maps observations to a finite set of classes (or *hypotheses*). In the case of classifying graphs by connectivity, the relevant hypothesis test has classes "the graphs which are not so connected" and "the graphs which are highly connected," which will be denoted H_0 and H_1, respectively, in Chapter 4.



(a) Hypothesis Testing without Control.

(b) Hypothesis Testing with Control.

Figure 2.1: Conventional sensing versus controlled sensing (figure courtesy of Sirin Nitinawarat).

The hypothesis testing problem without control (conventional sensing) is shown in Fig. 2.1a. In this framework, we observe a function of an underlying state X, $g(X)$. Based on $g(X)$, we form an estimate of the underlying information state, $\hat{X}$, to perform inference. In the *controlled sensing* paradigm, the experimenter is allowed to select a control U to change the observation model (but not the underlying information state, as in the case of conventional control theory) in order to maximize the useful information of the observation and having observations drawn from $g(X, U)$. By selecting

the control U appropriately, provably good inference (asymptotic error decay in the number of samples) can be achieved [16].

An illustrative example of controlled sensing is disease diagnosis. Assume the underlying information state X is either "the patient has the disease" or "the patient does not have the disease." The doctor diagnosing the disease can choose to perform different techniques (controls U) to help diagnose the disease, such as blood tests or exploratory surgeries. The distribution of observations is a function of the underlying information state and the chosen technique. Note that not all techniques will be equally useful in differentiating the classes for the underlying information state. In the case of identifying if the patient has a sore throat, two possible controls the doctor could apply are looking at the patient's throat or testing his or her knee reflexes. Since looking at the patient's throat gives more information about sore throats than do knee reflexes (because knee reflexes are approximately independent of throat condition), the doctor elects to observe the underlying information state through the throat to better perform inference. A conventional sensing paradigm such as using patient testimony on internet forums can be less effective than the active diagnosis of a controlled sensing paradigm.

2.1 Fixed Sample-Size Testing

The fixed sample-size hypothesis testing problem amounts to a hypothesis test between vectors of observations. We begin with a Bayesian approach to this problem. Let $\mathbf{Y} = (Y_1, \ldots, Y_N)^T$ be a random vector of length N drawn from distribution p_i for some $i \in [M]$ representing a sample of size N. Note that this model allows for general joint distributions between samples. The primary references for this section are [17, 18, 19].

2.1.1 Simple Hypothesis Tests

The fixed sample-size hypothesis testing problem is to decide between

$$H_i : \mathbf{Y} \sim p_i \qquad i \in [M] \tag{2.1}$$

from a vector of observations $\mathbf{y}$. Such a test is said to be *simple* since there is one possible distribution under each hypothesis.

We define the *likelihood ratio* between and j as

$$L_j(\mathbf{y}) = \frac{p_j(\mathbf{y})}{p_0(\mathbf{y})} \qquad j \neq 0 \tag{2.2}$$

For the case where $M = 2$, we denote L_1 as L. The likelihood ratios are a *sufficient statistic* for detection; that is, they capture all of the information from $\mathbf{y}$ necessary for distinguishing between the hypotheses. A *decision rule* is a mapping $\delta : \mathbf{y} \to [M]$ which declares a hypothesis from a vector of observations. Let $\Gamma_i = \delta^{-1}(\{i\})$ denote the *decision region* corresponding to hypothesis i. Note that $\Gamma = \cup_i \Gamma_i$ is the set of all possible observation vectors.

We define the *Bayes risk* associated with δ with prior distributions π_j and costs $\{C_{ij}\}_{i,j=1}^{M}$ as

$$R(\delta) = \sum_{i,j \in [M]} C_{ij} P_j[\Gamma_i] \pi_j \tag{2.3}$$

This quantity can be naturally interpreted as a risk because the i,j-th summand is the average cost associated with declaring i as the true hypothesis when j is the true hypothesis.

In this section, we will consider the case where $M = 2$ unless otherwise noted. We say δ is optimal if δ minimizes $R(\delta)$. It can be shown [18] that the structure of the optimal decision rule when $M = 2$ amounts to thresholding the likelihood ratio:

$$\delta(\mathbf{y}) = \begin{cases} 1 & \text{if } L(\mathbf{y}) \geq \eta \\ 0 & \text{if } L(\mathbf{y}) < \eta \end{cases} \tag{2.4}$$

for some threshold η determined by the costs and priors. A case where this test is useful is in communication systems to detect a bit, where H_0 is "0 was sent," while H_1 is "1 was sent." It is reasonable to use the prior $\pi_0 = \pi_1 = \frac{1}{2}$ since the bit should be approximately uniformly distributed after pre-transmission compression [20]. The dependence on the priors can be troublesome in applications where they are not accurately known, such as in system monitoring, where under H_0 is the system is working properly and under H_1 is the system is failing. A mismatched prior for system monitoring can increase delays in failure detection or increase the number of false alarms,

which can have serious economic impact for the system operator. The $M > 2$ case amounts to forming $M-1$ linear combinations of the $M-1$ likelihood ratios and comparing them to thresholds. Details are given in Section 2.7 of [18] along with examples of detecting a quadrature phase-shift keying signal in Gaussian noise.

In the same vein, we can also consider the more general *likelihood-ratio test* (LRT) δ_{LRT}:

$$\delta_{\text{LRT}}(\mathbf{y}) = \begin{cases} 1 & \text{if } L(\mathbf{y}) > \eta \\ 1 & \text{w.p. } \gamma \text{ if } L(\mathbf{y}) = \eta \\ 0 & \text{if } L(\mathbf{y}) < \eta \end{cases} \tag{2.5}$$

This test structure[1] can be shown to be optimal under some useful criteria for appropriate choices of η and γ determined by the costs, priors, and constraints for optimization.

One such criterion is the *minimax* criterion, which minimizes the worst-case Bayes risk over all possible priors for LRTs – that is,

$$\delta_M = \arg\min_{\delta} \max_{0 \le \pi_0 \le 1} R(\delta; \pi_0) \tag{2.6}$$

where $R(\delta; \pi_0)$ denotes the Bayes risk for π_0. This is useful when the priors are not known accurately, and can be shown via a convex optimization argument as in [18].

Another useful criterion is the Neyman-Pearson (NP) criterion. Define the probability of detection (also known as the *power*) as $P_D(\delta) = P_1(\Gamma_1)$ and the probability of false alarm (also known as the *size*) as $P_{\text{FA}}(\delta) = P_0(\Gamma_1)$. The NP test is the LRT δ which maximizes $P_D(\delta)$ subject to $P_{\text{FA}}(\delta) \le \alpha$ where $\alpha \in (0, 1)$ is a design constraint on the false alarm probability. That is,

$$\delta_{\text{NP}} = \max_{\delta : P_{\text{FA}}(\delta) \le \alpha} P_D(\delta) \tag{2.7}$$

A proof of existence, uniqueness, and false alarm constraint being met with equality of this test can be found in [17]. The constraint on α is an user-specified design choice, and can be motivated through a failure monitoring

[1] If L has no point masses under either hypothesis, we can ignore the randomized decision when $L(\mathbf{y}) = \eta$, because this occurs on a set of probability measure 0.

scenario. Consider H_0 to be the system is functioning normally, while H_1 is the system malfunctioning. When the system is malfunctioning, it must be diagnosed at a high cost to the operator in time and labor. A trivial test which guarantees high detection probability is $\delta = 1$ (that is, always say the system is malfunctioning). However, if failures are rare, then this test leads to a high operation cost for the plant. By specifying α appropriately, one can trade off between detecting the failure and the cost of mis-diagnosing a failure.

2.1.2 Composite Hypothesis Tests

We will now consider the case of composite hypothesis testing where $M = 2$. Composite hypothesis tests allow $\mathbf{Y}$ to be drawn from a distribution in a parameterized family $\{p_\theta\}_{\theta \in \Lambda_i}$ under each hypothesis. That is,

$$H_i : \mathbf{Y} \sim p_\theta \qquad \theta \in \Lambda_i, i = 0, 1 \tag{2.8}$$

Typically, Λ_i will be some subset of an Euclidean space.

Note that the simple hypothesis case described in Section 2.1 is when Λ_i are singleton sets. Several hypothesis tests are possible in this scenario, of which we will describe the *uniformly most powerful* (UMP) test and the *generalized likelihood ratio* (GLR) test, both of which do not rely on priors. The reader is referred to [17] for a description of *locally most powerful tests*.

The composite Bayesian case reduces to the Bayesian case for simple hypotheses. Let the prior distributions $\pi_i(\theta)$ for H_i assign probabilities to each $\theta \in \Lambda_i$. Then, defining $p_i(\mathbf{y}) = \int_{\theta \in \Lambda_i} p_\theta(\mathbf{y}) \pi_i(\theta) d\theta$, we can see that the composite hypothesis test is equivalent to the simple hypothesis test between $H_i : \mathbf{Y} \sim p_i$.

The uniformly most powerful test uses the Neyman-Pearson theory to design a test whose structure is independent of Λ_0, Λ_1 that maximizes the power of the test under a size constraint. This test is not guaranteed to exist, but when it exists is an α-level NP test with an LRT structure between p_{θ_0} and p_{θ_1} where $\theta_i \in \Lambda_i$ for all choices of θ_i. That is,

$$\delta_{\text{UMP}} = \underset{\delta : P_{\text{FA}}(\delta; \theta_0) \leq \alpha \forall \theta_0 \in \Lambda_1}{\arg\max} P_D(\delta; \theta_1) \quad \forall \theta_1 \in \Lambda_1 \tag{2.9}$$

A problem which admits a UMP test distinguishing between a sample of a Gaussian distribution of known variance having mean zero or positive mean. However, if this problem is modified to distinguish between zero mean or non-zero mean, a UMP test does not exist because the α-level NP test structure is different if the test is between zero mean and positive or negative means. A sufficient criterion for UMP test existence is the monotone likelihood ratio criterion [18]. Note that the complexity of this computation is just the complexity of an LRT and is thus computationally efficient.

The generalized likelihood ratio test relies on the statistic

$$T_{\mathrm{GLR}}(\mathbf{y}) = \frac{\sup_{\theta_1 \in \Lambda_1} p_{\theta_1}(\mathbf{y})}{\sup_{\theta_0 \in \Lambda_0} p_{\theta_0}(\mathbf{y})} \tag{2.10}$$

This statistic can be approximately interpreted as finding the *maximum likelihood estimate* of θ_i under hypothesis i (that is, the θ_i that maximizes the probability of observing $\mathbf{y}$ under that hypothesis), $\hat{\theta}_i$, and calculating the likelihood ratio between $p_{\hat{\theta}_0}$ and $p_{\hat{\theta}_1}$. That is, the GLR statistic is a likelihood ratio between the most likely distributions under either hypothesis.

The GLR test structure is given by

$$\delta_{\mathrm{GLR}}(\mathbf{y}) = \begin{cases} 1 & \text{if } T_{\mathrm{GLR}}(\mathbf{y}) > \eta \\ 1 & \text{w.p. } \gamma \text{ if } T_{\mathrm{GLR}}(\mathbf{y}) = \eta \\ 0 & \text{if } T_{\mathrm{GLR}}(\mathbf{y}) < \eta \end{cases} \tag{2.11}$$

for some design parameter η. While error analysis is typically not analytically tractable and calculation of the statistic requires solving optimization problems, the GLR test has sufficient simplicity in calculation and good error performance to make it useful for applications even when priors are known [19].

Note the graph classification problem proposed in this thesis is a composite hypothesis test, but the requirement of control to specify a sampling policy as in Chapter 4 will be seen as the natural formulation.

2.1.3 Asymptotics of Fixed Sample-Size Tests

Particularly in the case of Chernoff's procedure, we will discuss performance of hypothesis tests in the asymptotic regime. For fixed sample-size tests, this will be as $N \to \infty$ and the components of $\mathbf{Y}$ are drawn independently and from identical distributions (i.i.d). This discussion is useful for several reasons. In cases when many samples are collected, asymptotic analysis can give simple analytical approximations and mathematically tractable error characterizations (which are not always exactly possible, such as in the GLR test case). It also provides a comparison point with sequential hypothesis testing. We will focus on the case where $M = 2$ and ignore randomization, since it will be invoked less often as the sample-size increases. All results are presented *almost surely*, or up to a set of probability zero.

Let the components of $\mathbf{Y}$ be drawn from p_0 under H_0 and p_1 under H_1. Define $L(Y_i) = \frac{p_1(Y_i)}{p_0(Y_i)}$. In this case, $L(\mathbf{Y}) = \prod_{i=1}^{N} L(Y_i)$. Define $S_N = \frac{1}{N}\sum_{i=1}^{N} \log L(Y_i)$. Then, the LRT can be written as

$$\delta_{\text{LRT}}(\mathbf{Y}) = \begin{cases} 1 & \text{if } S_N \geq \eta(N) \\ 0 & \text{if } S_N < \eta(N) \end{cases} \tag{2.12}$$

where $\eta(N)$ is a threshold of the LRT chosen as a function of N such that $\eta(N)$ converges to η (so that the threshold for detection does not grow indiscriminately). By the strong law of large numbers, under H_1, $S_N \to D(p_1||p_0)$ and under H_0, $S_N \to -D(p_0||p_1)$ where

$$D(p||q) = \int_x p(x) \log \frac{p(x)}{q(x)} \, d\mu(x) \tag{2.13}$$

is the *Kullback-Leibler (KL) divergence (or distance)* between distributions p and q [20]. The KL divergence appears in information theory as a natural way to measure distances[2] between distributions and is described further in [20]. In particular, $D(p||q) \geq 0$ with equality if and only if $p = q$. Thus asymptotically, the probability of false alarm and missed detection (given by $P_M(\delta) = P_1(\Gamma_0)$) both go to zero if $\eta \in (-D(p_0||p_1), D(p_1||p_0))$. The theory of *large deviations* allows us to quantify the rate at which this convergence

[2]This is not a true distance, as it is not symmetric. However, it possesses sufficiently many properties of metrics to give a useful information geometry.

occurs. We will sketch the ideas for bounding the rate and refer the interested reader to Section 3.2 of [18] for more details. We define the *large deviations rate function* of the i.i.d. sequence $\{\log Y_i\}$ as

$$I_0(z) = \max_{u \in \mathbb{R}} (uz - \log(E_0[e^{u \log Y_i}])) \tag{2.14}$$

which is a concave maximization problem [21]. Cramer's theorem [18, 22] states

$$\lim_{n \to \infty} \frac{1}{N} \log P_{\text{FA}}(N) = -I_0(\eta) \tag{2.15}$$

This states that the false alarm probability decays exponentially at rate $I_0(\eta)$ with sample-size up to a subexponential factor in sample-size. A similar analysis shows that the $P_M(N)$ also decays exponentially at rate $I_0(\eta) - \eta$ with sample-size up to a sub-exponential factor in sample-size. Given priors π_0, π_1, defining the error probability as $P_E = \pi_0 P_{\text{FA}} + \pi_1 P_M$, it can be shown that $\eta = 0$ makes the error probability decay most rapidly under a uniform cost assignment (that is, the cost of making an error is 1 and correct decision is 0). Choosing η to be slightly above $-D(p_0||p_1)$ can be interpreted analogously to the Neyman-Pearson test, while choosing η to be slightly below $D(p_1||p_0)$ is analogous to the Neyman-Pearson test where P_M is constrained rather than P_{FA} (these are sometimes referred to Type I and Type II NP tests, respectively) [18]. This result will serve as motivation for the sequential probability ratio test (SPRT) to sequentially distinguish between two simple hypothesis in Section 2.2.

Some results on the asymptotics of the GLR test are given in [19, 18]. No useful results are known in the general case, but good error properties are inherited from the asymptotics of maximum-likelihood estimation [18, 23].

2.2 Sequential Probability Ratio Test

The sequential probability ratio test (SPRT) was introduced by Abraham Wald [24] to test between two simple hypotheses in a non-fixed sample-size setting. In Section 2.1.3, we showed that a sequence of likelihood ratio tests gives exponential missed detection and false alarm probability decays as a function of the KL-distances between two hypothesis while the sample-size tends to infinity. The SPRT modifies this idea by sequentially deciding how

many samples to use for a decision as well as how to declare a hypothesis as a function of these observations. We will assume $M = 2$ and samples under H_i are drawn i.i.d. from distribution p_i and the likelihood ratio is defined as $L(y) = \frac{p_1(y)}{p_0(y)}$. Some extensions to $M > 2$, collectively referred to as MSPRTs, were proposed by Baum and Veeravalli in [25], and were shown to have strong asymptotic error decay guarantees. These extensions are omitted due to space constraints.

A *sequential test* consists of three parts: a *stopping rule*, which is a collection of functions indexed by time, $\{\phi_n\}$, a *stopping time*, N, which is the smallest time when the test elects to stop collecting samples, and a *decision rule*, which is a collection of functions indexed by time $\{\delta_n\}$. $\phi_n : \mathbf{y}^n \to \{0, 1\}$ is a function from the first n observations to $\{0, 1\}$ and indicates to continue sampling (0) or stop sampling and make a decision (1). The stopping time N is the smallest time such that $\phi_n = 1$. At time N, the test stops, and a decision based on $\mathbf{y}^N$ is made using the decision rule $\delta_N : \mathbf{y}^N \to [M]$ to declare a particular hypothesis [18].

The algorithmic description of the SPRT is given in Algorithm 1. From the algorithmic description, we can read off the sequential test structure. The loop specifies $\phi_n = \mathbb{1}_{S_n \notin (A,B)}$, and the if statement specifies $\delta_n = \mathbb{1}_{S_n \geq B}$. We will see a similar structure for Chernoff's test and controlled sensing. As shown in Section 2.1.3, $\frac{S_n}{n}$ will converge almost surely to $-D(p_0||p_1)$

Algorithm 1 Sequential Probability Ratio Test [17]

```
Specify finite thresholds A < 0, B > 0.
while S_n = Σ_{i=1}^{n} log L(y_i) ∈ (A, B) do
    take samples
end while
if S_N ≤ A then
    stop and declare H_0
else
    stop and declare H_1
end if
```

under H_0 or to $D(p_1||p_0)$ under H_1, so it is reasonable to expect that a decision will almost surely be made in finite time by the SPRT. The *Wald-Wolfowitz theorem* shows the SPRT is optimal in the sense that among all sequential tests which achieve a certain P_M, P_{FA}, the SPRT minimizes the expected number of samples under H_0 and H_1, similar to the NP test in

the fixed sample-size setting. Taking a fixed sample-size test as a special case of a sequential test, we see that the SPRT uses less samples on average for the same error performance. These results follow from Wald's identity which is a result for stopping times on certain martingales. Some details are provided in [17]. The most important consequence of Wald's identity is $E[S_N] = E[N]E[S_1]$ when $N < \infty$ a.s. [26]

The *Wald approximations* give simple rules for specifying A, B to meet target P_{FA}, P_M constraints, and are derived from Wald's identity and the assumption $S_N \approx A$ or B. In particular, setting $A = \frac{P_M}{1-P_{\text{FA}}}, B = \frac{1-P_M}{P_{\text{FA}}}$ for small P_M, P_{FA} ensures that the constraints on error probabilities are approximately met [17].

While the SPRT has many advantages over the fixed sample-size tests described in Section 2.1, analysis heavily depends on the i.i.d assumption, and $E[N] < \infty$ does not guarantee short stopping times. Short stopping times can be guaranteed by truncating the test after a fixed number of samples leading to the *truncated SPRT*, while retaining good error performance for sufficiently late truncation. The SPRT also does not handle sequential composite hypothesis testing, which can lead to bad results under model mismatch versus a composite fixed sample-size test [17].

2.3 Chernoff's Procedure and Controlled Sensing

Controlled sensing [27, 28] is a framework for multiple hypothesis testing with causal observation control in the fixed sample-size and sequential settings. We will only consider the sequential setting because it forms the basis of Chapter 4. Controlled sensing extends the work of Chernoff [16] in the simple hypothesis case. The binary composite hypothesis case is similar. Consider the sequential simple hypothesis test where

$$H_i : Y_k \text{ i.i.d} \sim p_i \text{ for some } p_i \in \{p_\theta\}_{\theta \in \Lambda_i}$$

where $i \in [M], k \in \mathbb{N}$. At each time, a control action $u \in [U]$ is selected, and induces a distribution on the observations from underlying hypothesis i as p_i^u. In contrast to most dynamical systems, the control does not affect the process observed, but changes the quality of observations. The goal of

controlled sensing is to select an optimal control for quick discrimination between the hypotheses subject to good asymptotic error decay. This allows for simple test designs to meet performance requirements. In particular, the tests designed via controlled sensing exhibit asymptotically optimal error decay with sample-size in a sense to be defined at the end of this section. This notion of optimality is used instead of the exact optimality stated for the SPRT due to intractability of analysis and because the large sample-size regime is a natural limit associated with low cost of sampling.

To account for the control, we define a *sequential test with control* [27] as a triple (ϕ, N, δ). ϕ is a (random) *causal observation control policy*, which takes on values from $[U] \cup \{S\}$ where $[U]$ are possible observation controls to continue sampling and S denotes a stop sampling control. N is the *stopping time* for the test. δ is the *decision rule*, which maps from the vector of controls up to time N, $\mathbf{u}^N$, and past observations up to time N, $\mathbf{y}^N$, into $[M]$. We denote the probability distributions of the causal observation control policy by $q(\cdot)$. Let $\hat{i}_k = \arg\max_{i\in[M]} p_i(y^k, u^k)$ denote the maximum likelihood estimate of the underlying hypothesis at time k, which is a function of observations and controls up to time k. The *(modified) Chernoff procedure* for controlled sensing is given in Algorithm 2. Note that this algorithm is

Algorithm 2 (Modified) Chernoff Procedure [27]

1: c is a design parameter. $\alpha > 1$ is a design parameter for modified Chernoff procedure
2: **while** $\min_{j\neq\hat{i}_n} \log\left(\frac{p_{\hat{i}_n}(\mathbf{y^n},\mathbf{u^n})}{p_j(y^n,u^n)}\right) < -\log c$ **do**
3: Draw sample from p^{u_n}
4: Draw control U_{n+1} from distribution

$$\begin{aligned} q(u_{n+1}) &= q(u_{n+1}|\hat{i}_n) \\ &= \arg\max_{\bar{q}(u)} \min_{j\in[M]-\hat{i}_n} \sum_u \bar{q}(u) D(p^u_{\hat{i}_n} || p^u_j) \end{aligned}$$

5: *(Modified Chernoff procedure only)* If $n = \lceil \alpha \rceil^l$ for $l \in \mathbb{N}$, let $q(u_n)$ be the uniform distribution on $[U]$ instead.
6: **end while**
7: Stop and declare $\hat{i}_N$

very similar to that of the SPRT. In the SPRT, the control actions are to continue sampling or stop sampling, versus a distribution on methods to continue sampling in controlled sensing. Thus, taking $U = 1$, $M = 2$, and

the hypotheses to be simple, we essentially recover the SPRT with thresholds $\pm \log c$.

In the case where $M = 2$, the optimization problem to select the control has an interpretation as a two player zero-sum game [16], where one player tries to pick a distribution of controls to maximize the average KL-distance to the alternative hypothesis while the other player picks the minimizing alternative hypothesis. Let the value of this game be denoted by $V = \max_{\bar{q}(u)} \min_{j \in [M]-\{\hat{i}_n\}} \sum_u \bar{q}(u) D(p^u_{\hat{i}_n} || p^u_j)$.

Chernoff's procedure applied to simple hypotheses from [16] is identical to Algorithm 2 in the case where $M = 2$ where the modified Chernoff procedure step is omitted. When applied to composite hypotheses, the minimization in the selection of q^* is over distributions under the hypothesis which is not $\hat{i}$. A key assumption in the original procedure is $D(p^u_i || p^u_j) > 0$ and $E_{p^u_i}[(\log(\frac{p^u_i}{p^u_j}))^2] < \infty$ for all $i \neq j$, $i, j \in [M]$ and $u \in [U]$. That is, the distributions induced by a control under any distinct hypotheses are distinct on a set of positive measure, and the log likelihood ratios between them have finite first and second moments.

We now describe the sense in which Chernoff's procedure is asymptotically optimal. Assume c is small (so the number of samples used by the test is large). Chernoff's procedure can be shown to be asymptotically optimal through the following results [16]: The expected sample-size is bounded above by $\frac{-(1+o(1)) \log c}{V}$, which follows from a bound on the stopping time, and the error probability is $O(c)$ which follows from standard integral bounding techniques. Combining these facts, we see the risk of this test is at most $\frac{-(1+o(1))c \log c}{V}$. It can also be shown for any test under the $D(p^u_i || p^u_j) > 0$ and finite second moment conditions stated previously such that the risk is $O(-c \log c)$ under any underlying hypothesis must have risk of at least $\frac{-(1+o(1))c \log c}{V}$ for any underlying hypothesis. The proof of this result relies on some martingale inequalities and integral bounds. Combining these results shows that any test which does significantly better than Chernoff's procedure for some underlying hypothesis must be significantly worse for some other underlying hypothesis. The asympotic risk decay of Chernoff's procedure can be used analogously to the Wald approximations for the SPRT to set test parameters, as it provides an order of error decay for setting sample-sizes in the presence of control.

In the case of the controlled sensing test, where $D(p^u_i || p^u_j) = 0$ is permissi-

ble, the proofs given in [16] essentially hold up to a result that $\hat{i}_n$ is strongly consistent (that is, $\hat{i}_n$ converges to the true hypothesis almost surely). In particular, if $\hat{i}_n$ is strongly consistent, then for sufficiently large (random) sample-sizes, T, $\hat{i}_n$ is the underlying hypothesis for all $n \geq T$. It can be shown under Chernoff's assumptions that if T is the smallest sufficiently large sample-size, then the probability that T exceeds n decays exponentially with n. The key ideas to relax Chernoff's assumption are that random sampling forces consistency and arbitrarily quick polynomial decays are possible (i.e., $O(n^{-s})$ for $s \in \mathbb{R}^+$) by appropriate choice of α close to 1. The use of polynomial decays is sufficient to modify Chernoff's proof to reach a similar conclusion in the M-ary case.

The case of $M = 2$ with composite hypotheses is treated in [16]. Some practical issues with Chernoff's procedure are outlined in Section 7 of [16]. A key issue is initial controls may have an undesirable distribution due to lack of information about the underlying hypothesis. In particular, the controls may only be useful if the hypothesis estimate is sufficiently close to the underlying hypothesis. This can be done by pre-loading some observations into the algorithm. The computational complexity of the optimization problem can also be undesirable even in the binary case, especially when U is large [29]. However, approximations or modeling simplifications can partially alleviate this drawback, as seen in Chapter 4.

CHAPTER 3

PRIOR WORK IN GRAPH SAMPLING AND INFERENCE

In this chapter, we provide an overview of some prior work in sampling and performing inference on graphs. The primary techniques used in practice are variants of random walks over graphs. The bulk of this chapter is dedicated to describing random walks. However, we first discuss two other techniques, random node selection and random edge selection. We will assume all observations are perfect when sampling in this chapter. That is, the vertex set is known and no missed or spurious edges are observed. The *budget* of a sampling scheme is the number of samples collected. A sampling scheme for a graph must trade off between complexity of the algorithm, the amount of edges stored in the sample (which corresponds to the level of resolution in the network), and the structure kept by the algorithm. We say a sampling scheme is *weighted* if it does not follow a uniform distribution.

3.1 Random Node and Edge Sampling

Random node sampling (or *uniform sampling* when the distributions involved are uniform) is the simplest way to sample a graph. The sample is formed by selecting a set of nodes at random, and letting the estimate of the graph be the set of edges which share an endpoint with the selected set of nodes. Since we focus on connectivity in this thesis, random node sampling is not a desirable method of acquiring samples from a graph because it does not use any graph structural constraints (such as following edges like a random walk) to preserve connectivity in order to collect samples, especially in the uniform sampling case. In fact, this method does not preserve much graph structure such as the degree distribution of nodes [6]. However, this technique has successfully been tuned to sample nodes with a given structure, such as

using the PageRank statistic (used by the Google search engine) as weights[1] [30, 6]. Note that uniform sampling is computationally cheap since it requires only a uniform random number generator and a fixed sampling budget. Non-uniform random node sampling can be computationally expensive depending on the desired probability distribution. The sampling portion of the controlled sensing scheme proposed in this thesis can be viewed as an intelligent variant of random node sampling to capture a particular form of structure in the presence of noise.

Random edge sampling is analogous to random node sampling, where a subset of edges is randomly selected rather than a subset of nodes. Aside from the identification of edges versus nodes being less of a realistic scenario (because the edges are typically what is desired to be learned), this technique can give excessively sparse samples of a graph, since the number of possible edges in a graph with N vertices grows as $\binom{N}{2} \in O(N^2)$. As with random node sampling, no structural characteristics of the graph are used to collect samples [6]. This technique can be viewed as random node sampling on the *line graph*, which is a dual to a graph [9].

3.2 Random Walks

The current state of the art techniques are random walks on graphs. A *random walk* is a random process for sampling the nodes a graph, where the $(n+1)$-th node sampled is selected from the neighbors of the n-th node sampled according to some probability distribution on the neighborhood of the n-th node. Since the random walk always samples neighbors, it follows paths within the graph and thus preserves much of the underlying graph structure. For example, if one is sampling a social network to study the propogation of a message, the mechanism which the message propogates is similar to the sampling procedure of a random walk. A drawback, which is immediate from the definition of a random walk, is the lack of reachability of all nodes in a disconnected graph. To alleviate this, one can construct a *random walk with jumps*, which follows a random walk sampling rule with probability $1-\lambda$ and a uniform sampling rule with probability λ. The choice

[1]The PageRank algorithm itself is a random walk algorithm, but the statistic used to weight the random walk can be adapted for weighted random node sampling.

of λ trades off between *exploration* of the graph outside the neighborhood or several edges away from the current node (through uniform sampling) and following the local graph structure (through the random walk). Empirical evidence suggests $\lambda = 0.15$ is a good value [6]. By appropriately constructing the probability distribution of the random walk, one can bias the random walk towards particular structures, such as weighting the random walk to explore more high (or low) degree nodes. Note that random walk-based techniques are computationally inexpensive. That is, given that many graphs in practice are sparse, the complexity associated with acquiring a sample is $O(\max_{i \in V} d_i) \ll N$.

3.3 Frontier Sampling

Frontier sampling (FS) is a random walk-based algorithm which attempts to preserve more structure than a random walk with jumps, while avoiding the pitfall of being stuck in one component. Frontier sampling was proposed in [7] and serves as a comparison point to the controlled sensing scheme proposed in Chapter 4. We present the centralized version of the algorithm, and note that it can be implemented as a distributed algorithm to sample very large graphs.

We first state the frontier sampling algorithm:

Algorithm 3 Frontier Sampling [7]

1: Fix sampling budget $B > 0$, $m \geq 1$ number of random walkers. Start with a graph estimate $\hat{G}$ to be the empty graph on vertex set V.
2: Choose $L = \{v_1, \ldots, v_m\}$, a set of m vertices, according to a uniform distribution.
3: **while** $B > 0$ **do**
4: Choose a vertex $v \in L$ according to the probability distribution $P(v_i \text{ is selected}) = d_{v_i} / \sum_{j \in L} d_{v_j}$.
5: Choose a neighbor of v uniformly,u, and replace v with u in L.
6: Add edge e_{uv} to $\hat{G}$.
7: B:=B-1
8: **end while**

In essence, frontier sampling starts m random walks, and at each iteration, randomly chooses a random walk to follow according to weights determined by the degree of the nodes currently visited by each random walk. It can

be proved that frontier sampling is equivalent to a random walk over the m-fold graph product [9] of the graph to be sampled with itself. Numerical demonstrations in [7] show that the FS algorithm gives better results than m independent random walkers or a single random walk for quantities such as degree distributions, especially in the tail of the distribution. Note that since this algorithm was designed purely with sampling in mind, its goal is to produce a small representative sample of the graph (with respect to number of edges) while ignoring fine grained details of the network structure. Since the graph sample produced by FS is small, the sample can be easily analyzed and archived. A simple modification to the FS algorithm is to retain all edges emanating from vertex v instead of just edge e_{uv} at the cost of additional storage. We use this modification to the FS algorithm as a fair comparison point to the algorithm designed in Chapter 4 rather than the conventional FS algorithm, because the sampling budget is then measured in vertex samples, as in the proposed controlled sensing algorithm, rather than in edge samples.

3.4 Detection of Nodes with Large Degree

In [31], a procedure is proposed for sampling a graph inspired by the PageRank algorithm [30] in order to infer a list of the k nodes of largest degrees in the graph. The sampling portion of the procedure consists of a random walk with jumps combined with a stopping rule based on the number of times nodes have been visited. The inference portion of the procedure maintains a list of the k nodes sampled by the sampling procedure with the largest degrees. This procedure is of interest, since it uses a sequential procedure to determine the sampling budget as well as some level of graph inference through a list of the k largest degree nodes. It should be noted that the random walk used to acquire samples does not re-weight given observation history as does the proposed controlled sensing test of Chapter 4. Thus, the sampling scheme is effectively independent of the inference scheme when the sampling budget is ignored.

CHAPTER 4

GRAPH CLASSIFICATION VIA CONTROLLED SENSING

This chapter of the thesis is an expanded version of Ligo, Atia, and Veeravalli [32].

4.1 Overview of Contributions

We pose the problem of classifying a graph by connectivity (measured through average node degree) by sampling nodes in the presence of noise (missing or spurious observed edges) as a composite sequential hypothesis test with controlled sensing, as described in Chapter 2 and [16, 27, 33]. The average node degree of a graph was described in Section 1.3.

In contrast to prior work, mainly from the social network literature overviewed in Chapter 3 and [31, 7, 6] where performance is only quantified on experimental data sets, such as the DBLP authorship graph in [31], the proposed framework allows for classification of graphs with a provably low number of samples when the classification error is desired to be low. While [6, 7] considered a fixed sampling budget procedure, the procedure developed herein sequentially determines the number of samples needed to classify the graph. Typically, a sequential hypothesis test results in a lower average number of samples used than a fixed sample budget test with the same error probability. The control used for graph sampling proposed in this chapter finds a favorable trade-off between *exploring* the graph and *exploiting* knowledge of the graph in order to improve the quality of graph observations for classification.

Also, unlike prior work where each edge is assumed to be fully observable and no edges which are absent from the graph are observed [6, 7], we consider more general graph observation models, where both real edges and edges not in the graph ("spurious edges") are probabilistically observed. The algorithm

proposed in this chapter handles imperfect observations of graphs and reduces to the fully observable model as a limiting case.

We first describe a controlled sensing test to classify graphs by average node degree where edges are probabilistically observable. Then, we compare the controlled sensing test to a random walk-based technique, frontier sampling (FS) [7], on a graph with probabilistic edge observation. When no spurious edges in the graph are observable, it is shown that the controlled sensing test outperforms FS with respect to error probabilities for a given number of samples in the low and medium edge observation probability regime. The controlled sensing test is also demonstrated on an observation model that allows for spurious edges with respect to different levels of spurious and true edge observation.

4.2 Graph Classification as a Sequential Hypothesis Test

Interactions (or connections) between nodes are described through the edges of a fixed underlying graph $G = (V, E)$ with sets $V = V_G$ and $E = E_G$ denoting the vertex (node) and edge sets, respectively. The underlying graph for the example considered in this section consists of the blue edges in Fig. 4.1.

Without loss of generality, we will assume that there are $N > 1$ vertices in the graph G and $V = [N]$ by a simple relabeling of vertices. In the controlled sensing paradigm, G is the underlying information state.

Define the classes of graphs $\mathcal{G}_0$ and $\mathcal{G}_1$ as

$$\begin{aligned} \mathcal{G}_0 &= \{G : \quad |V| = N, \bar{d}_G \leq \eta\} \\ \mathcal{G}_1 &= \{G : \quad |V| = N, \bar{d}_G > \eta\} \end{aligned} \tag{4.1}$$

where $\bar{d}_G$ was defined in (1.5). The class $\mathcal{G}_0$ represents graphs which "are not so connected," while $\mathcal{G}_1$ represents graphs which are "highly connected" when the average node degree is compared to a user-specified threshold η. Note that $\bar{d}_G$ is a non-negative integer multiple of $\frac{1}{N}$ and the problem is only of interest for $0 < \eta < N-1$ (where the bounds correspond to the empty and complete graphs, respectively). To classify a graph based on connectivity, we

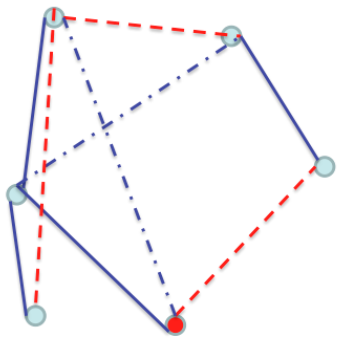


Figure 4.1: An example of a graph observed under OM2. The red vertex (i) is the observed vertex and the edges observed are the dashed red edges and solid blue edges incident to it. The solid blue edges represent edges in the graph which could be observed at the observation time (each independently with probability p). The dashed blue edges represent edges in the graph which are not observable at the observation time (each independently with probability q). The red dashed edges are some of the possible spurious edges – edges which are observable, but are not in the underlying graph (each independently with probability q).

can pose the problem as the composite binary hypothesis testing problem,

$$H_0 : G \in \mathcal{G}_0$$
$$H_1 : G \in \mathcal{G}_1$$

In order to distinguish these hypotheses with an underlying information state (graph) G, we propose two different observation models of the underlying graph called *Observation Model 1* and *Observation Model 2*. An observation model for the underlying graph is G is a collection of probability distributions $\{P_G^u\}_{u \in V}$, where P_G^u specifies the distribution of observed edges incident to u when node u is observed. The definition of observation models specifies the control set in the controlled sensing paradigm to be

$$\mathcal{U} = V \bigcup \{S\} \tag{4.2}$$

where $v \in V$ is the control to collect a sample by observing node u in the graph and S denotes the control to stop taking samples and make an estimate of the class of the underlying graph.

First, we define *Observation Model 1* (OM1). In this model, when node

i is observed, each edge in $\{e_{ij} : j \in N_G(i)\}$ is observed with probability p independent of the others. Hence, if node i is selected ($U = i$), the observation $Y \subseteq \{e_{ij} : j \in N_G(i)\}$ of edges connected to i is drawn according to the probability mass function (pmf) $P_G^i(y)$, where

$$P_G^i(y) = p^{|y|}(1-p)^{d_i - |y|} \tag{4.3}$$

A similar model, *Observation Model 2* (OM2), allows for spurious edges to be observed. In this case, if node i is observed, the observations are a subset of all possible edges incident to node i partitioned as the disjoint union $A \cup B$. A is a subset of $\{e_{ij} : j \in N_G(i)\}$ for which each edge is observed with probability p independently of the others (corresponding to true interactions or edges) and B is a subset of $\{e_{ij} : j \in N_{G^C}(i)\}$ where each edge is observed with probability $q < p$ (corresponding to spurious interactions or edges) independent of others. Thus, the observations follow the pmf $P_G^i(y)$, where

$$P_G^i(y) = q^{|y \cap E^C|}(1-q)^{((N-1)-d_i)-|y^C \cap E^C|} p^{|y \cap E|}(1-p)^{d_i - |y^C \cap E|} \tag{4.4}$$

Note this model reduces to OM1 as $q \searrow 0$ where we take $0^0 = 1$. OM2 can also be viewed as using OM1 on both G and G^C independently with probabilities p and q, respectively. Note if $p < q$, we can switch the roles of G and G^C to reach this definition of OM2 and perform classification on G^C to conclude the connectivity of G. The problem is not interesting when $p = q$ because every underlying graph has the same distribution of observations (since every possible edge is observable with probability p) for all nodes and thus the classes are not distinguishable. An example of OM2 is given in Fig. 4.1. Ignoring the red edges gives an example of OM1.

A functional interpretation of p in OM1 and OM2 is the partial observability of interactions in the underlying graph. If one is monitoring a network such as a telephone network, one may only see interactions when there are calls between connected nodes giving rise to a $p < 1$. The q parameter allows for uncertainty in observations as a sort of noise floor, such as calls between people who should not be connected in the telephone network. The q parameter provides extraneous information about the underlying graph G. The uncertainty about G induced by q provides an additional level

of robustness to the observation model. Note the conventional algorithms described in Chapter 3 implicitly assume $p \approx 1$ and $q \approx 0$ and thus operate approximately under OM1 with $p = 1$.

4.3 Sequential Test

In order to classify graphs, we pose the problem as a controlled hypothesis testing problem (as in Chapter 2 or [33, 28, 16]) where the controls from $\mathcal{U}$ select which node should be observed in order to maximize the quality of the observations for classification or to stop collecting samples. First, we recall the components of a controlled hypothesis testing problem.

Let u^n and y^n denote the list of controls and observations from time 1 to n respectively. If the test stops at time n, i.e., if $U_n = S$, we make a decision $\delta(y^n, u^n) \in \{0, 1\}$ about the hypothesis. Hence, the sequential test $\gamma = \{\phi_k, \tilde{N}, \delta\}$ consists of a control policy

$$\phi_k : \mathcal{U}^{k-1} \times \mathcal{Y}^{k-1} \to \mathcal{U},\ k = 0, 1, \ldots \tilde{N} - 1$$

where $\mathcal{U}$ is specified in (4.2) and

$$\mathcal{Y}_i = \{e_{U_i l} : l \in V\} \subseteq V \times V - \{(v, v) : v \in V\}$$

is the set of allowable observations at time i, a stopping rule with stopping time

$$\tilde{N} = \inf\{n > 0 : U_n = S\}$$

and a decision rule

$$\delta_{\tilde{N}} : \mathcal{U}^{\tilde{N}-1} \times \mathcal{Y}^{\tilde{N}-1} \to \{0, 1\}$$

The test is designed to minimize the expected stopping time (number of vertices sampled) for the best order of decay of the error probabilities with sample size.

Thus, by designing the parameters described in the test appropriately, the proposed test will give desirably low rates of error classification with sufficiently many samples.

Let $\mathcal{G}_i$ be the hypothesis which contains $\widehat{G}$, the estimate of the graph G, and $\mathcal{G}_j$ the alternative hypothesis which does not contain $\widehat{G}$.

We propose the following controlled sensing sequential test for classifying graphs based on connectivity:

1. **Graph Estimation:** At each time k, find the maximum-likelihood estimate (MLE) of G, $\widehat{G} = \widehat{G}(y^k, u^k)$.

2. **Estimate of Hypothesis:** Find $\widehat{i}(k)$, the estimate of the hypothesis at time k, which is 1 if $\bar{d}_{\widehat{G}} > \eta$ and 0 otherwise.

3. **Stopping Rule:** The controller stops at time k and declares $\widehat{i}(k)$ if

$$\min_{\widetilde{\boldsymbol{G}}:\widetilde{G}\in\mathcal{G}_j} \log \frac{P_{\widehat{G}}(y^k, u^k)}{P_{\widetilde{G}}(y^k, u^k)} > \log \beta \tag{4.5}$$

   where $P_G(y^k, u^k)$ is the joint distribution of the observations and the controls for underlying graph G induced by the observation model $P_G^u(y)$ and the causal control distributions $q(u_k|u^{k-1}, y^{k-1})$ specified by the control policy. The graph $\widetilde{G}$ is the nearest graph under the alternative hypothesis. Thus, the left-hand side (LHS) of (4.5) is simply the likelihood ratio of the joint distributions given the current graph estimate and the nearest graph in the alternative hypothesis. If OM1 is used, it is sufficient to stop if $\widehat{i}(k) = 1$. This is due to $\bar{d}_{\widehat{G}}$ being a monotone property of $\widehat{G}$ and by construction of the MLE under OM1, $\widehat{G}(y^{k-1}, u^{k-1}) \subseteq \widehat{G}(y^k, u^k)$ due to the lack of spurious observed edges.

   The parameter β is a design threshold representing the required confidence (measured by log-likelihood ratio) for the maximum likelihood estimate of the graph's class being sufficiently separated from the alternate class to declare a decision.

   Control Policy: If the decision is to continue sampling, the controller chooses a control action U_{k+1} drawn from the distribution

$$q_{k+1}^*(u) \triangleq \mathbb{P}\{U_{k+1} = u|\widehat{I}_k = \widehat{i}\}$$

   where the probability vector $\boldsymbol{q}_{k+1}^*$ is obtained as a solution to the following max-min optimization problem

$$\max_{q(u),u\in V} \min_{\widetilde{G}:\widetilde{G}\in\mathcal{G}_j} \sum_{u=1}^{N} c(u, u^k, \widehat{G}) q(u) D(P_{\widehat{G}}^u, P_{\widetilde{G}}^u) \tag{4.6}$$

where $D(P_1, P_2)$ denotes the Kullback-Leibler (KL) distance between the distributions P_1 and P_2 defined in (2.13) with μ as the counting measure (that is, we replace the integral with a sum) and $c(u, u^k, \widehat{G})$ is a user-designed positive weighting function whose purpose will be discussed in Section 4.3.1.

If the KL distance between distributions is zero under at least one control, we modify the test by using a uniform control at times $\lceil a^l \rceil$ for $l \in \mathbb{N}$ and $a > 1$ fixed. By [27], this test has asymptotically optimal error decay with sample size under OM2 with $0 < q < p < 1$ when the $c(\cdot)$ function is constructed appropriately (such as when $c(\cdot)$ is equal across all choices of nodes to sample after a finite number of samples). Under OM1, the experimental results show promise, but asymptotically optimal error decay with sample size is not guaranteed by [27]. When $c(\cdot)$ is a constant, this is the standard Chernoff procedure from [16].

A simple modification to this procedure is to truncate the test after a fixed number of samples (representing a maximum allowable vertex sampling budget), akin to the truncated sequential probability ratio test (SPRT) to sequentially decide between two hypotheses with independent and identically distributed observations. In the case of the truncated SPRT, the test retains good stopping times and error performance (for suitably large number of samples) while avoiding pitfalls such as sample paths where a large number of samples are needed to make a decision [18].

There are two simple extensions to sampling $1 < m < N$ nodes at each time when each node sampled at the same time has independent observations (which is a reasonable approximation for sparse graphs where $m \ll N$). The first extension is where the m nodes are not necessarily distinct, m controls are drawn from the same distribution and applied at step 3 of the test. This requires calculating one control policy for m observations, and is a reasonable approximation of the control policy stated above when $\boldsymbol{q}$ does not change rapidly with each sample. The exact solution would require enumeration over all $\binom{N+m-1}{m}$ m-multisets of V as a control policy which is not computationally feasible.

The second extension is the case of m distinct nodes observed at each time. The control is defined on the $\binom{N}{m}$ distinct labeled m-subsets of V. To obtain the control policy in this case, the matrix M' is replaced by a matrix with $\binom{N}{m}$

rows where the i-th row of this matrix is the sum of the rows of M' indexed by the i-th subset. Due to the size of this problem, it is likely unreasonable to use for moderate N and m. It is unlikely the prior procedure would introduce repeated observations or significant deviation from the original procedure in these cases as well, and is thus preferable. In both cases, the graph estimate and the stopping rule are identical.

4.3.1 Control Policy

The optimization problem in (4.6) can be viewed as a two-player zero sum game. Let $\widehat{G} \in \mathcal{G}_i$. Then, player 1, the maximizer, tries to choose a distribution $\boldsymbol{q}$ over the vertex set V, while player 2 chooses the nearest graph in $\mathcal{G}_j$ for $j \neq i$. We will show that this can be done by inserting (resp. removing) edges to (resp. from) $\widehat{G}$ if $\widehat{G} \in \mathcal{G}_0$ (resp. $\mathcal{G}_1$). Since a graph can have $O(N^2)$ edges on N vertices, the number of edges which can be inserted or removed from $\widehat{G}$ can be significantly larger than N. In particular, when η is chosen sufficiently high and the underlying graph is sufficiently sparse, there exist enough edges in G^C (G) such that the average node degree can be increased (decreased) from that of $\widehat{G}$ so that player 1 is forced to adopt a uniform control. By good design of $c(\cdot, u^k, \widehat{G})$, the uniformity of control can be alleviated by trading exploration with exploitation. Let $\widehat{\boldsymbol{d}}$ and $\widetilde{\boldsymbol{d}}$ be the degree vectors of the graphs $\widehat{G}$ and $\widetilde{G}$, respectively. In the case of OM1, if $N_u(\widehat{G}) \subseteq N_u(\widetilde{G})$,

$$\begin{aligned} D(P^u_{\widehat{G}}, P^u_{\widetilde{G}}) &= \sum_{i=0}^{\widehat{d}_u} \binom{\widehat{d}_u}{i} p^i (1-p)^{\widehat{d}_u - i} \log \left(\frac{p^i (1-p)^{\widehat{d}_u - i}}{p^i (1-p)^{\widetilde{d}_u - i}} \right) \\ &= \alpha'_u (\widetilde{d}_u - \widehat{d}_u) \log \frac{1}{1-p} \end{aligned}$$

where α'_u is 1 if $\widehat{d}_u > 0$ and is 0 otherwise. If $N_u(\widehat{G}) \not\subseteq N_u(\widetilde{G})$, we have $D(P^u_{\widehat{G}}, P^u_{\widetilde{G}}) = \infty$ since every edge observed is known to be in the underlying graph.

Thus, under OM1, (4.6) can be written as:

$$\max_{q(u), u \in V} \min_{\widetilde{G}: \widetilde{G} \in \mathcal{G}_j} \sum_{u=1}^{N} \alpha_u q(u) (\widetilde{d}_u - \widehat{d}_u) \tag{4.7}$$

where $\alpha_u = \alpha'_u c(u, u^k, \widehat{G})$.

In practice, it can be useful to replace α'_u with

$$\alpha'_u = \begin{cases} 1 & \text{if } \widehat{d}_u > 0 \\ f(u, u^k, \widehat{G}) & \text{o.w.} \end{cases}$$

where f is an experimenter-designed non-negative weighting function chosen to penalize unexplored nodes and encourage exploiting explored nodes. While the choice of $f(u, u^k, \widehat{G})$ and $c(u, u^k, \widehat{G})$ does not change the asymptotic error performance under appropriate conditions (such as equalizing over u over a finite number of samples), a good choice will improve non-asymptotic error performance.

We begin with the case where $\widehat{G} \in \mathcal{G}_0$ under OM1. If $\widehat{G} \in \mathcal{G}_1$, then the test would have stopped under OM1. It is necessary that the minimizer in (4.6) contain $\widehat{G}$, else there exist vertices which have infinite KL distances contrary to the minimizer's objective. Thus, we see that we can think of the minimizer as being formed by inserting edges into $\widehat{G}$ to reach $\mathcal{G}_0$.

The optimization problem is easily posed in terms of the incidence matrix of $\widehat{G}^C$, $M_{\widehat{G}^C}$. Let $M'_{\widehat{G}^C}$ denote the matrix obtained from $M_{\widehat{G}^C}$ by multiplying the i-th row of $M_{\widehat{G}^C}$ by $\alpha_i, i = 1, \ldots, N$. P_N is the N-dimensional probability simplex and $IS = \{\boldsymbol{x} \in \{0,1\}^{|E_{\widehat{G}^C}|} : \boldsymbol{x} \text{ has } \lceil |\eta - \bar{d}_{\widehat{G}}| \frac{N}{2} \rceil \text{ non-zero entries}\}$ (where IS stands for "insertion set" and the i-th entry of a vector in IS corresponds to edge i in a listing of the edges $E_{\widehat{G}^C}$ to be inserted into $\widehat{G}$ to form the minimizer).

Note that

$$\widetilde{\boldsymbol{d}} - \widehat{\boldsymbol{d}} = M_{\widehat{G}^C} \boldsymbol{x} \tag{4.8}$$

where $\widetilde{G}$ corresponds to the graph consisting of the edges in $\widehat{G}^C$ specified by $\boldsymbol{x}$ and containing $\widehat{G}$.

Hence, (4.7) can be written as

$$\max_{\boldsymbol{q} \in P_N} \min_{x \in IS} \boldsymbol{q} M'_{\widehat{G}^C} \boldsymbol{x} \tag{4.9}$$

That is, when player 1 picks a distribution, player 2's policy is to insert edges in $\widehat{G}$ until the new graph is in $\mathcal{G}_1$. Player 2 does so by adding edges which do not exist in $\widehat{G}$ whose endpoints are of lowest sum weight, akin to the optimization for the stopping rule shown later. Since each edge inserted into

a graph on N vertices increases the average node degree by $\frac{2}{N}$, $\lceil |\eta - \bar{d}_{\widehat{G}}| \frac{N}{2} \rceil$ edges must be inserted into $\widehat{G}$ to get a graph in $\mathcal{G}_1$.

A satisfactory relaxation of (4.9) for computational purposes is

$$\max_{\boldsymbol{q} \in P_N} \min_{\{\boldsymbol{x} \in \mathbb{R}^{|E_{\widehat{G}^C}|} : \boldsymbol{x} \geq 0, \mathbf{1}^\top \boldsymbol{x} = \lceil |\eta - \bar{d}_{\widehat{G}}| N/2 \rceil\}} \boldsymbol{q} M'_{\widehat{G}^C} \boldsymbol{x} \tag{4.10}$$

The relaxation follows by first relaxing the constraint that the components of $\boldsymbol{x}$ are in $\{0, 1\}$ to lying in the interval $[0, 1]$, and then relaxing the box constraints to a non-negativity constraint. Based on simulations, it seems the box constraints are rarely active. There are more precise linear programming relaxations which retain the box constraints, but these relaxations introduce many auxiliary variables when compared to (4.12). As the control policy must be computed for each sample, the increased number of variables incurs a high computational cost for the proposed algorithm.

A simple functional interpretation can be assigned to (4.10) as a two-player zero sum game by substituting $\boldsymbol{u} = \lceil |\eta - \bar{d}_{\widehat{G}}| N/2 \rceil \boldsymbol{x}$. Since we are only concerned in the distribution $\boldsymbol{q}$, this problem is equivalent to

$$\max_{\boldsymbol{q} \in P_N} \min_{\{\boldsymbol{u} \in \mathbb{R}^{|E_{\widehat{G}^C}|} : \boldsymbol{u} \geq 0, \mathbf{1}^\top \boldsymbol{u} = 1\}} \boldsymbol{q} M'_{\widehat{G}^C} \boldsymbol{u} \tag{4.11}$$

by ignoring a factor of $\frac{1}{\lceil |\eta - \bar{d}_{\widehat{G}}| N/2 \rceil}$ in the objective function. From this, we see that $\boldsymbol{u}$ is a probability vector, as is $\boldsymbol{q}$. Thus, we can interpret the solution $\boldsymbol{q}$ as finding a saddle point in a two-player zero sum game in mixed strategies, where player 1 (the experimenter) picks a distribution of vertices to sample and player 2 (nature) picks a distribution of edges to insert into $\widehat{G}$ to confuse player 1.

Equation (4.11) can be rewritten to the equivalent linear program (LP):

$$\max_{\boldsymbol{q}, v} v \tag{4.12}$$
$$\text{subject to: } \begin{cases} \sum_u q_u [M'_{\widehat{G}^C}]_{uj} \geq v & j = 1, \ldots, |E_{\widehat{G}^C}| \\ \sum_u q_u = 1, \quad 0 \leq q_u \leq 1 \end{cases}$$

using standard LP techniques (see, for example, Section 11.3 of [34]) and solved using standard LP solvers in polynomial time in N (though efficiency will depend on the solver) [21]. For OM1, (4.12) completely specifies the control policy as $\widehat{G} \notin \mathcal{G}_1$.

The case for OM2 is similar. The necessary results are derived in Appendix B. First, we consider $\widehat{G} \in \mathcal{G}_0$. Comparing (B.9) with (4.9), we see the equations are identical when $M'_{\widehat{G}^C}$ is replaced with $\widetilde{\widetilde{M}}_{\widehat{G}^C}$ as defined in (B.7). Thus, the control policy can be calculated with (4.12) with the same substitution. When $\widehat{G} \in \mathcal{G}_1$, we see (B.12) and (4.9) are identical when $M'_{\widehat{G}^C}$ is replaced with $\widetilde{\widetilde{M}}_{\widehat{G}}$ as defined in (B.7) and IS is replaced with $\{\boldsymbol{x} \in \{0,1\}^{|E_G|} : \boldsymbol{x} \text{ has } |E_{\widehat{G}}| - \lfloor \frac{N}{2}(\eta - \bar{\boldsymbol{d}}_{\widehat{G}}) \rfloor \text{ non-zero entries}\}$. Thus, we can use an LP similar to (4.9) to calculate the control policy.

4.3.2 Maximum-Likelihood Graph Estimation

In this section, we propose a simple MLE of G. By using a more advanced estimators which captures more realistic graph structures (such as when G is drawn from some generative model), we can improve classification performance at a higher computational cost.

Under OM1, it is clear that the MLE of the graph, $\widehat{G}$, is simply the graph consisting of all edges observed up to the current time (since no edge observed is spurious). Thus, we consider OM2.

At time k and for all $i \in V$, define $\mathcal{T}_i(k) = \{j \in \{1, \ldots, k\} : U_j = i\}$ as the set of all times up to k when node i is selected. We assume that the observations of the various nodes are independent, conditioned on their respective neighborhoods

$$P(y^k|G) = \prod_{i=1}^{N} P(y_{\mathcal{T}_i(k)}|N_i(G)) \tag{4.13}$$

where $y_{\mathcal{T}_i(k)} = \{y_j : j \in \mathcal{T}_i(k)\}$.

Define $\mathcal{T}_{ij}(k) = \mathcal{T}_i(k) \cup \mathcal{T}_j(k)$. This is the number of times edge e_{ij} can be *potentially observed* up to time k. Denote the number of times edge e_{ij} is actually observed up to time k by $l_{ij}(k)$. If $e_{ij} \in E$ (resp. $e_{ij} \notin E$), the probability of the observation sequence is $p^{l_{ij}(k)}(1-p)^{|\mathcal{T}_{ij}(k)|-l_{ij}(k)}$ (resp. $q^{l_{ij}(k)}(1-q)^{|\mathcal{T}_{ij}(k)|-l_{ij}(k)}$).

Thus, $\widehat{G}$ at time k is specified by

$$[A_{\widehat{G}}]_{ij} = \begin{cases} 1 & \text{if } i \neq j \text{ and } p^{l_{ij}(k)}(1-p)^{|\mathcal{T}_{ij}(k)|-l_{ij}(k)} > q^{l_{ij}(k)}(1-q)^{|\mathcal{T}_{ij}(k)|-l_{ij}(k)} \\ 0 & \text{otherwise} \end{cases} \tag{4.14}$$

The MLE can be calculated in $O(N)$ time by keeping track of $\mathcal{T}_{ij}$ and noting at each time, $N-1$ edges have to have their estimates updated.

For the case where m nodes are observed at each time, we can simply apply (4.14) in any order to the observed nodes in $O(mN)$ time (since at most $m(N-1)$ edges must have their $\mathcal{T}_{ij}$ sets and thus estimates update).

4.3.3 Stopping Rule

The form of (4.5) under OM1 is simple since $\widehat{G} \in \mathcal{G}_0$:

$$\begin{aligned} \log \frac{P_{\widehat{G}}(y^k, u^k)}{P_{\widetilde{G}}(y^k, u^k)} &= \sum_{j=1}^{k} \log \frac{P_{\widehat{G}}(y_j, u_j)}{P_{\widetilde{G}}(y_j, u_j)} \\ &= -\log(1-p) \sum_{j=1}^{k} (\widetilde{d}_{u_j} - \widehat{d}_{u_j}) \\ &= -\log(1-p) \sum_{i=1}^{N} (\widetilde{d}_i - \widehat{d}_j) |\mathcal{T}_i(k)| \end{aligned}$$

Collecting constants gives the stopping rule

$$\min_{\widetilde{G} \in \mathcal{G}_1, E_{\widetilde{G}} \supset E_{\widehat{G}}} \sum_{i=1}^{N} (\widetilde{d}_i - \widehat{d}_j) |\mathcal{T}_i(k)| > \log \beta \tag{4.15}$$

The optimization problem in (4.15) can be solved $O(N^2 \log(N^2))$ time by noting that we can find the minimizer by inserting edges into $\widehat{G}$, and adding edge e_{ij} increases the sum by $|\mathcal{T}_i(k)| + |\mathcal{T}_j(k)| = |\mathcal{T}_{ij}(k)|$ independent of the other edges (so insertion order does not matter). $E_{\widehat{G}^C}$ and $|\mathcal{T}_{ij}(k)|$ can be calculated by finding the non-diagonal zeros of $A_{\widehat{G}}$ and updating $\mathcal{T}_{ij}(k)$ after every sample. To find $\widetilde{G}$, sort the $O(N^2)$ edges by $|\mathcal{T}_{ij}(k)|$ in ascending order in $O(N^2 \log(N^2))$ time and insert the first $\lceil |\eta - \bar{d}_{\widehat{G}}| N/2 \rceil$ edges into $\widehat{G}$. Thus, the LHS of the stopping rule is the sum of the $\lceil (\eta - \bar{d}_{\widehat{G}}) \frac{N}{2} \rceil$ smallest values of $|\mathcal{T}_{ij}(k)|$.

We can update a sorted list of $|\mathcal{T}_{ij}|$ at time k for time $k+1$ in $O(N \log N)$ by maintaining a sorted list of $|\mathcal{T}_{ij}(k)|$ for $e_{ij} \in E_{\widehat{G}^C}$, removing the $O(m(N-1))$ edges added to $\widehat{G}$ at $k+1$ when m nodes are sampled at once and updating the $O(m(N-1))$ values of of $|\mathcal{T}_{ij}(k+1)|$ which changed after sampling via binary search and an external table of pointers to the $|\mathcal{T}_{ij}|$. Thus, once the stopping rule has been calculated, at future times it can be calculated in $O(N \log N)$ time. An extension of this strategy is also proposed for OM2.

For OM2, we note

$$\log \frac{P_{\widehat{G}}(y^k, u^k)}{P_{\widetilde{G}}(y^k, u^k)} = \log \left(\frac{\prod_{e_{ij} \in E_{\widehat{G}}} p^{l_{ij}(k)} (1-p)^{|\mathcal{T}_{ij}(k)| - l_{ij}(k)}}{\prod_{e_{ij} \in E_{\widetilde{G}}} p^{l_{ij}(k)} (1-p)^{|\mathcal{T}_{ij}(k)| - l_{ij}(k)}} \times \frac{\prod_{e_{ij} \in E_{\widehat{G}^C}} q^{l_{ij}(k)} (1-q)^{|\mathcal{T}_{ij}(k)| - l_{ij}(k)}}{\prod_{e_{ij} \in E_{\widetilde{G}^C}} q^{l_{ij}(k)} (1-q)^{|\mathcal{T}_{ij}(k)| - l_{ij}(k)}} \right) \tag{4.16}$$

First, note (4.16) is non-negative since $\widehat{G}$ is a graph which maximizes $P_G(y^k, u^k)$ over G. If $\widetilde{G} = \widehat{G}$, then (4.16) is 0. Note that the numerator of (4.16) is independent of $\widetilde{G}$. Also, the contribution of edge e_{ij} to (4.16) is independent of all other edges and only depends on the number of times e_{ij} was observed and could potentially be observed. From this, we can see that under $\widehat{G} \in \mathcal{G}_0$, the $\widetilde{G}$ which minimizes (4.16) must satisfy $\widehat{G} \subset \widetilde{G}$. If an edge e_{ij} is present in $\widehat{G}$ but not in $\widetilde{G}$, we have the ratio of the corresponding terms in the numerator and denominator of (4.16) is $\frac{p^{l_{ij}(k)}(1-p)^{|\mathcal{T}_{ij}(k)| - l_{ij}(k)}}{q^{l_{ij}(k)}(1-q)^{|\mathcal{T}_{ij}(k)| - l_{ij}(k)}} > 1$ by (4.14). Including e_{ij} in $\widetilde{G}$ increases the average node degree of $\widetilde{G}$ and reduces (4.16) as the ratio of the terms corresponding to e_{ij} in (4.16) is 1, thus decreasing (4.16). Thus, $\widehat{G} \subset \widetilde{G}$ under $\widehat{G} \in \mathcal{G}_0$. A similar argument shows under $\widehat{G} \in \mathcal{G}_1$, $\widetilde{G} \subset \widehat{G}$. By the independence of contributions to (4.16) for each edge, we can see the minimizer can be calculated by starting with $\widetilde{G} = \widehat{G}$ and inserting (resp. removing) edges if $\widehat{G} \in \mathcal{G}_0$ (resp. $\widehat{G} \in \mathcal{G}_1$) and the minimizer does not insert (resp. remove) more edges than necessary to have $\widetilde{G} \in \mathcal{G}_1$ (resp. $\widetilde{G} \in \mathcal{G}_0$). This insight is similar to that in Appendix B.

Define the change in (4.16) of adding edge $e_{ij} \in \widetilde{G}^C$ to $\widetilde{G}$ as δ_{ij} where

$$\delta_{ij} = \begin{cases} l_{ij}(k) \log \frac{q}{p} + (|\mathcal{T}_{ij}(k)| - l_{ij}(k)) \log \frac{1-q}{1-p} & \text{if } |\mathcal{T}_{ij}(k)| > 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.17}$$

If $e_{ij} \in \widetilde{G}$, then $-\delta_{ij}$ is the change in (4.16) when e_{ij} is removed from $\widetilde{G}$.

We first consider $\widehat{G} \in \mathcal{G}_0$. As argued previously, to find the minimizing $\widetilde{G}$ in (4.16), we need to add edges to $\widehat{G}$, and the order in which we add the edges does not matter. Start with $\widetilde{G} = \widehat{G}$. If we add an edge to $\widetilde{G}$, it corresponds to moving the edge from $\widetilde{G}^C$ to $\widetilde{G}$. In order to have $\widetilde{G} \in \mathcal{G}_1$, we must add $\lceil (\eta - \bar{d}_{\widehat{G}})\frac{N}{2} \rceil$ edges to $\widetilde{G}$. The edges added to $\widetilde{G} = \widehat{G}$ in order to find the minimizer of (4.16) are the $\lceil (\eta - \bar{d}_{\widehat{G}})\frac{N}{2} \rceil$ edges in $E_{\widehat{G}^C}$ with the smallest δ_{ij} for $e_{ij} \in E_{\widetilde{G}^C}$ and, thus, the minimum value of (4.16) is the sum of the $\lceil (\eta - \bar{d}_{\widehat{G}})\frac{N}{2} \rceil$ smallest δ_{ij} for $e_{ij} \in E_{\widehat{G}^C}$. Ties in edge selection can be broken arbitrarily.

When $\widehat{G} \in \mathcal{G}_1$, we remove $\lfloor \frac{N}{2}(\eta - \bar{d}_{\widehat{G}}) \rfloor$ edges from $\widehat{G}$ to find the $\widetilde{G} \in \mathcal{G}_0$ minimizing (4.16). The edges removed are the $\lfloor \frac{N}{2}(\eta - \bar{d}_{\widehat{G}}) \rfloor$ edges in $\widehat{G}$ with the smallest $-\delta_{ij}$ (and the minimum value of (4.16) is the sum of these $-\delta_{ij}$ values). Ties in edge selection can be broken arbitrarily.

One can calculate all δ_{ij} in $O(N^2)$ time and sort them in $O(N^2 \log(N^2))$ time. Note that we only need to update $N-1$ (resp. at most $m(N-1)$ if m nodes are sampled at once) δ_{ij} after each sample. Thus, a sorted list of δ_{ij} for each $\widehat{G}$ and $\widehat{G}^C$ can be updated in $O(N \log N)$ time via binary search and insertion into a sorted list with an external table of pointers to δ_{ij}.

In practice, it is useful to start the algorithm with some initial observations of each node (or a subset of nodes) in order to reduce the stopping time as in Remark 7.1 of [16].

4.4 Numerical Results

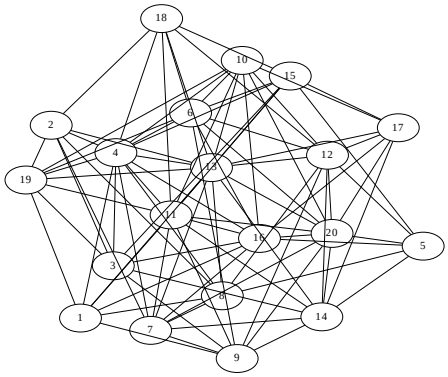


Figure 4.2: Graph G with 20 nodes with average node degree 8.9.

For concreteness and conciseness, we present classification performance on

an Erdös-Rènyi (ER) model A generated graph with uniform edge probability $\frac{1}{2}$ on 20 nodes with average node degree 8.9 shown in Fig. 4.2. The construction of ER model A graphs was described in Section 1.2. ER graphs are of interest from a performance analysis perspective because proving properties of an appropriate family of ER graphs shows that the property holds for almost all graphs. As per Remark 7.1 in [16], the procedure presented is of interest when η is close to $\bar{d}_G$. Thus, we show results for $\eta = 8.8$ and $\eta = 9.0$ with the tests truncated to 1000 samples. $c(u, u^k, \widehat{G})$ is the number of times a node has been sampled up to time k, or 1 if it has not been sampled, i.e.,

$$c(u, u^k, \widehat{G}) = \begin{cases} |\mathcal{T}_u(k)| & \text{if } |\mathcal{T}_u(k)| > 0 \\ 1 & \text{o.w.} \end{cases}$$

The standard deviation of all probabilities presented down to 10^{-3} is at least an order of magnitude below the probabilities. Comparison to the frontier sampling algorithm is performed by converting the frontier sampling edge budget into a vertex budget as discussed in Section 3.3. The results for OM1

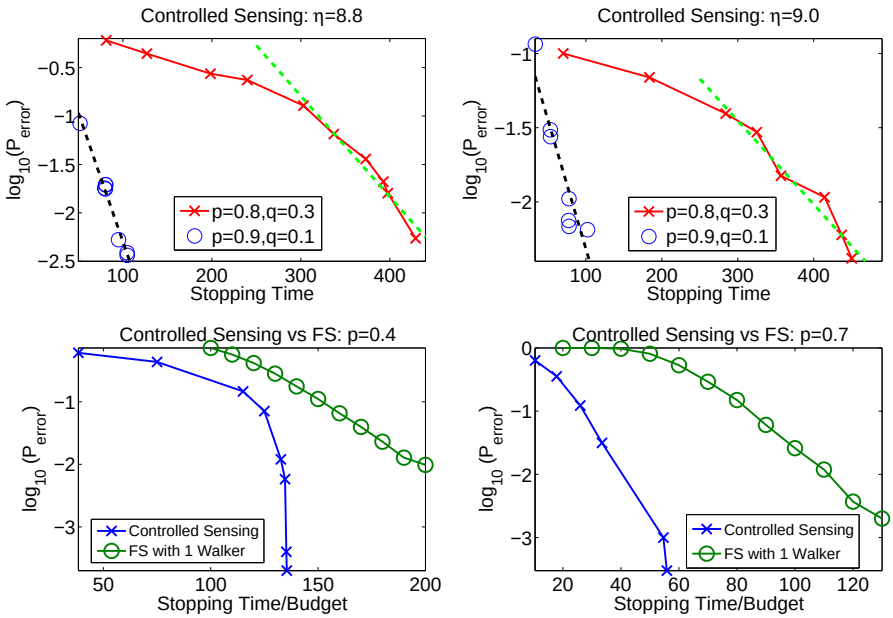


Figure 4.3: First Row: Controlled sensing with spurious observations. Second Row: Controlled sensing versus FS without spurious observations. Note that connected lines are drawn only for readability.

with $p = 0.4$ and $p = 0.7$ are given in the bottom row of Fig. 4.3. The

controlled sensing test with a given expected stopping time performs strictly better than the FS using the expected stopping time of the controlled sensing test with $\eta = 8.8$ (since false alarms are not possible under this model, this is the hardest value of η to classify) in the sense of lower error probabilities. The lower stopping times are in part due to the stopping rule, which accounts for p while FS assumes $p \approx 1$. The control is also tailored to capture the structure of G which controls the average node degree rather than the general structure of G as in the case of FS. There is also a threshold phenomena in detection, where the probability of error falls off at a very high rate when controlled sensing has (on average) observed enough edges to conclude the graph has average node degree greater than η. It was also found that FS offered little improvement under this graph model until the number of random walks used was on the order of N since it is unlikely then for a random walk to get trapped in a small neighborhood in the observed graph.

Under OM2, FS is not directly applicable due to the spurious edges, since a spurious edge allows a random walk to transition between non-neighboring nodes in G. Thus, we compare two controlled sensing tests to study the relative performance difference between tests with different observation probabilities for both true and spurious edges ($p = 0.8, q = 0.3$ and $p = 0.9, q = 0.1$) in the bottom row of Fig. 4.3. Lowering q and increasing p significantly reduces the number of samples needed to achieve a given error probability. The dashed least-squares fit lines shown for the tails of the data indicate that in these regimes the error probability decays approximately exponentially. This behavior is consistent with the asymptotic exponential decay of the error probability with the stopping time in Chernoff's procedure and controlled sensing [16, 27]. In particular, we see the approximate control policy in (4.12) and choice of $c(\cdot)$ do not have asymptotic penalities on order of error decay for the proposed classification algorithm relative to the standard controlled sensing approach. Quantifying the asymptotic rate of error decay is a difficult analytical problem since it depends on the value of the game specified by the control policy (which in turn depends heavily on the structure of the underlying graph).

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

In this thesis, we proposed a controlled sensing-based framework for classifying a graph with respect to average node degree as a measure of connectivity under noisy observations of missing or spurious edges. This framework provides a joint classification and sampling scheme for studying some properties of graphs in contrast to the independent sampling and inference techniques described in Chapter 3. From the controlled sensing theory, our framework inherits provably good classifiers. We validated the test for connectivity on a random graph and showed it outperformed random walk-based approaches at low target error rates. It is important to note this framework provides a theory which can be applied to other graph properties even though we have focused on connectivity in this thesis.

Extending this work will primarily be in the direction of forming good computational approximations and simulation techniques. In order to validate classifier performance for low error probabilities, techniques such as importance sampling [23] are required due to variance of error probability estimates. Deriving an importance sampling algorithm for simulating classifiers on graphs would help computational validation of the proposed controlled sensing test on larger graphs due to the increased number of samples required. The main challenge in this area is choosing the distribution for importance sampling. Since the control policy involves solving an optimization and all probability distributions in this work are discrete, choosing probability distributions which allow for importance sampling is difficult.

In contrast to the random walk techniques discussed in Chapter 3, whose computational complexity per sample collected is controlled by the maximum node degree (which is effectively a small constant in sparse graphs), our algorithm has computational complexity per sample collected as a polynomial in the number of vertices because of the control policy, stopping rule, and graph estimate calculations. For small graphs or where the cost of each

sample is expensive, the complexity per sample is acceptable. However, for large graphs, the computational complexity growth rate makes it prohibitively expensive to implement a controlled sensing test directly. One way to reduce the computational complexity of a controlled sensing test may be to solve the controlled sensing test *locally* by combining controlled sensing-based techniques with a random walk-like structure as follows: One can form a subgraph based on the past few nodes observed and their neighbors in the estimated graph and solve the controlled sensing controlled policy on this subgraph. The stopping rule would be represented by some accumulated statistic over the local problems akin to the log-likelihood ratio used in the full problem. Taking the subgraph to be only the prior node observed would lead to a random walk. One can also include a probabilistic random jumps to other nodes in the graph in order to avoid the random walk-like sampling being trapped in a small portion of the graph. This is similar to the random walk with jumps described in Chapter 3.

Another direction of interest is working with generative models of graphs, such as assuming G came from one of the models discussed in Chapter 1. It does not seem likely that assuming a generative model for the underlying graph will simplify complexity in a significant manner since parameter estimation is typically a combinatorial optimization problem. If this is accomplished, it is feasible to apply controlled sensing techniques on large datasets. For example, one could study the connectivity structure of a graph corresponding to a large social network (e.g., Facebook).

Another interesting area is developing distributed and parallelizable controlled sensing tests for graph classification, which can be useful for monitoring large networks such as the power grid. Applications such as monitoring the power grid may require other notions of connectivity or graph properties, which can be posed in this framework by redefining the classes of graphs. However, even among the measures of connectivity discussed in this thesis, having a computationally feasible way to calculate the graph property or optimize over a set of graphs constrained by this property may be difficult (as in the case of clustering coefficients and mean geodesic path length). Introducing more than two classes of graphs is another interesting extension, such as an extension to "not very connected," "somewhat connected," and "highly connected" classes where the "somewhat connected" class has graphs with average node degree between the other two classes.

Finally, another interesting direction is hypothesis testing of processes on graphs. Many system monitoring problems can be posed as quickest detection problems, where the distribution of observations changes at some unknown time and it is desired to detect the change occurring as quickly as possible after it has occurred. Developing a controlled sensing algorithm on graphs for quickest detection could be useful for monitoring power outages or failure mechanisms of complex systems where dependencies are efficiently modeled by a graph.

APPENDIX A

CALCULATION OF KL DISTANCE UNDER OM2

In this appendix, we outline the calculation of the KL distance $D(P_G^u, P_H^u)$ where G, H are graphs on the same vertices. Let a_p be the number of edges incident to u common to both G and H, a_q be the number of edges incident to u common to both G^C, H^C. Let b_p be the number of edges which are incident to u in G but not in H, and b_q be the number of edges which are incident to u in H but not in G. Thus, $a_p + b_p = d_{G,u}$ and $a_p + b_q = d_{H,u}$ where the subscript indicates a pair (graph, vertex) and all possible edges incident to u belong to exactly one of the sets counted by a_p, a_q, b_p, b_q. Since the probability of observing a subset of edges within each of the aforementioned classes depends solely on the size of the subset, the KL distance can be calculated by summing over subsets of all possible sizes dependent on which graph they are contained in:

$$D(P_G^u, P_H^u) =$$

$$\sum_{i=0}^{a_p}\sum_{j=0}^{a_q}\sum_{k=0}^{b_p}\sum_{l=0}^{b_q}\binom{a_p}{i}\binom{a_q}{j}\binom{b_p}{k}\binom{b_q}{l}p^{i+k}(1-p)^{a_p+b_p-(i+k)}q^{j+l}(1-q)^{a_q+b_q-(j+l)}$$

$$\log\left(\frac{p^{i+k}(1-p)^{a_p+b_p-(i+k)}q^{j+l}(1-q)^{a_q+b_q-(j+l)}}{p^{i+l}(1-p)^{a_p+b_q-(i+l)}q^{j+k}(1-q)^{a_q+b_p-(j+k)}}\right)$$

$$= \sum_{i=0}^{a_p}\sum_{j=0}^{a_q}\sum_{k=0}^{b_p}\sum_{l=0}^{b_q}\binom{a_p}{i}\binom{a_q}{j}\binom{b_p}{k}\binom{b_q}{l}p^{i+k}(1-p)^{a_p+b_p-(i+k)}q^{j+l}(1-q)^{a_q+b_q-(j+l)}$$

$$((k-l)\log p + (b_p - b_q + l - k)\log(1-p) + (l-k)\log q + (b_q - b_p - l + k)\log(1-q))$$

$$= \sum_{k=0}^{b_p}\sum_{l=0}^{b_q}\binom{b_p}{k}\binom{b_q}{l}p^k(1-p)^{b_p-k}q^l(1-q)^{b_q-l}$$

$$((k-l)\log p + (b_p - b_q + l - k)\log(1-p) + (l-k)\log q + (b_q - b_p - l + k)\log(1-q))$$

$$= \sum_{k=0}^{b_p}\sum_{l=0}^{b_q}\binom{b_p}{k}\binom{b_q}{l}p^k(1-p)^{b_p-k}q^l(1-q)^{b_q-l}\left((k-l)\alpha + ((b_p-b_q)+(l-k))\beta\right))$$

$$= (b_p p - b_q q)\alpha + ((b_p - b_q) + (b_q q) - (b_p p))\beta$$
$$= (p\alpha + (1-p)\beta)b_p + (-(q\alpha + (1-q)\beta))b_q \tag{A.1}$$

Where $\alpha = \log p - \log q$ and $\beta = \log(1-p) - \log(1-q)$. Note the coefficients of b_p, b_q are positive when $0 < q < p < 1$. Thus, the KL divergence only depends on the number of edges which are in G but not H incident to u and vice versa. Thus, adding an edge to G which is not in H (resp. removing an edge from G which is in H) increases the KL distance by $-(p\alpha + (1-p)\beta)$ (resp. $-(q\alpha + (1-q)\beta)$) and removing an edge from G which is not in H (resp. adding an edge to G which is in H) decreases the KL distance by $-(p\alpha + (1-p)\beta)$ (resp. $-(q\alpha + (1-q)\beta)$).

APPENDIX B

DERIVATION OF THE CONTROL POLICY UNDER OM2

In this appendix, we derive the form of the control policy under OM2. With some care, one can also derive the OM1 control policy in this manner though the results given in Section 4.3.1 give a simpler derivation without the need to track infinite quantities. The general form of the control policy is given in (4.6). Without loss of generality, assume η is irrational (since average node degrees are integer multiples of $\frac{1}{N}$, an irrational number between η and the next largest multiple of $\frac{1}{N}$ can be used instead without changing the graphs contained in $\mathcal{G}_0$ and $\mathcal{G}_1$).

First, note the edge set of any graph on vertex set $V = [N]$ can be partitioned into a subset of edges on graph H and on H^C for any $H = (V, E_H)$. Thus, for a graph $\widetilde{G}$ on vertex set V, we can write its incidence matrix with respect to another graph $\widehat{G}$ on vertex set V as

$$M_{\widetilde{G}} = \left[M_{\widetilde{G}\cap\widehat{G}} | M_{\widetilde{G}\cap\widehat{G}^C}\right] \tag{B.1}$$

where we define $G \cap H = (V, E_G \cap E_H)$.

Let $\boldsymbol{k} = \begin{bmatrix} D(P^1_{\widehat{G}}, P^1_{\widetilde{G}}) \\ D(P^2_{\widehat{G}}, P^2_{\widetilde{G}}) \\ \vdots \\ D(P^N_{\widehat{G}}, P^N_{\widetilde{G}}) \end{bmatrix}$

Let $\boldsymbol{x} \in \{0,1\}^{\binom{N}{2}}$ be a vector used to index graphs on V. The first $|E_{\widehat{G}}|$ coordinates will correspond to edges in $\widehat{G}$, while the remaining coordinates correspond to edges in $\widehat{G}^C$. For convenience, we will denote the incidence matrix of the complete graph (the graph on V with all possible edges) as M. By (B.1), we can write

$$M = \left[M_{\widehat{G}} | M_{\widehat{G}^C}\right] \tag{B.2}$$

Thus, we see if the j-th coordinate of $\boldsymbol{x}$ is 1, then the graph corresponding to $\boldsymbol{x}$ has the j-th column of M as one of the columns of its incidence matrix.

We will call this graph G_x.

Also,

$$\boldsymbol{d}_x = M\boldsymbol{x} \tag{B.3}$$

where $\boldsymbol{d}_x$ is the degree vector associated with the graph G_x.

From (B.2), we consider the matrix

$$\widetilde{M} = \left[\widetilde{M}_{\widehat{G}} | \widetilde{M}_{\widehat{G}^C}\right] = \left[-(p\alpha + (1-p)\beta)M_{\widehat{G}} | - (q\alpha + (1-q)\beta)M_{\widehat{G}^C}\right] \tag{B.4}$$

where α, β are as in (A.1). Note that $\widetilde{M}$ has the first submatrix being non-positive and the second submatrix being non-negative, with the non-zero entries the same as those of M.

Thus, defining $\boldsymbol{x}_{\widehat{G}}$ to be the $\boldsymbol{x}$ corresponding to $\widehat{G}$, we see

$$\boldsymbol{k} = \widetilde{M}\boldsymbol{x} - \widetilde{M}\boldsymbol{x}_{\widehat{G}} \tag{B.5}$$

since the first $|E_{\widehat{G}}|$ components of x correspond to the edges counted by b_p and a_p in (A.1) and the other components correspond to the edges counted by b_q. The subtraction removes the edges corresponding to a_p. Note that $\widetilde{M}\boldsymbol{x}_{\widehat{G}}$ is a constant for a given calculation of the control policy.

Thus, (4.6) becomes

$$\min_{q(u), u \in V} \min_{\boldsymbol{x} \in \{0,1\}^{\binom{N}{2}} : G_x \in \mathcal{G}_j} \sum_{u=1}^{N} c(u, u^k, \widehat{G}) q(u) D(P^u_{\widehat{G}}, P^u_{\widetilde{G}})$$

which is equivalent to

$$\min_{q(u), u \in V} \min_{\boldsymbol{x} \in \{0,1\}^{\binom{N}{2}} : G_x \in \mathcal{G}_j} \boldsymbol{q}\, \mathrm{diag}(c(1, u^k, \widehat{G}), c(2, u^k, \widehat{G}), \ldots, c(N, u^k, \widehat{G}))\widetilde{M}(\boldsymbol{x} - \boldsymbol{x}_{\widehat{G}}) \tag{B.6}$$

by (B.5) and writing the sum as an inner product.

We note that $\mathrm{diag}(c(1, u^k, \widehat{G}), c(2, u^k, \widehat{G}), \ldots, c(N, u^k, \widehat{G}))\widetilde{M}$ can be collected into one matrix $\widetilde{\widetilde{M}}$ where the i-th row consists of the i-th row of $\widetilde{M}$ multiplied by $c(i, u^k, \widehat{G})$. $\widetilde{\widetilde{M}}$ can be partitioned analogously to $\widetilde{M}$ to be

$$\widetilde{\widetilde{M}} = \left[\widetilde{\widetilde{M}}_{\widehat{G}} | \widetilde{\widetilde{M}}_{\widehat{G}^C}\right] \tag{B.7}$$

where $\widetilde{\widetilde{M}}_{\widehat{G}}$ and $\widetilde{M}_{\widehat{G}}$ have the same dimensions (as do $\widetilde{\widetilde{M}}_{\widehat{G}^C}$ and $\widetilde{M}_{\widehat{G}^C}$).

Thus, we can rewrite (B.6) via (B.7)

$$\min_{q(u),u\in V} \min_{\boldsymbol{x}\in\{0,1\}^{\binom{N}{2}}:G_x\in\mathcal{G}_j} \boldsymbol{q}\left[\widetilde{\widetilde{M}}_{\widehat{G}}|\widetilde{\widetilde{M}}_{\widehat{G}^C}\right]\begin{bmatrix}\boldsymbol{x}_{E_{\widehat{G}}}-\mathbf{1}_{|E_{\widehat{G}}|}\\ \boldsymbol{x}_{E_{\widehat{G}^C}}\end{bmatrix} \tag{B.8}$$

where we have partitioned $\boldsymbol{x}$ according to which edges in a graph on V are indexed by $E_{\widehat{G}}$ or $E_{\widehat{G}^C}$. It is clear that if $\boldsymbol{x}$ corresponded to $\widehat{G}$, then the objective function of the optimization problem would be zero identically (though by definition, G_x is in the alternate class of $\widehat{G}$).

We now consider the two cases:

1. $\widehat{G}\in\mathcal{G}_0$:

   In this case, the minimizer must find a graph with more edges than $\widehat{G}$ in the class $\mathcal{G}_1$. Let the minimizer be denoted G_x^*. If $\widehat{G}\not\subseteq G_x^*$, then the endpoints of $E_{G_x^*}-E_{\widehat{G}}$ will each have their KL divergences under OM2 between $\widehat{G}$ and G_x^* increased by the coefficient of b_p in (A.1) times the number of edges with that endpoint in $E_{G_x^*}-E_{\widehat{G}}$ (call this penalty A). The remaining edges of G_x^* must be from $E_{\widehat{G}^C}$. The edges in G_x^* which are not in $\widehat{G}$ is larger than if $\widehat{G}\subset G_x^*$ in order to meet the average node degree constraint of $\mathcal{G}_1$, and the endpoints of those edges will each have their KL divergences under OM2 between $\widehat{G}$ and G_x^* increased by the coefficient of b_q in (A.1) times the number of edges with that endpoint. Noting that in this case, more edges from $\widehat{G}^C$ are in G_x^* than if $\widehat{G}\subset G_x^*$. Thus, we see that the graph $G_x^{**}=\widehat{G}\cup G_X^*$ has $D(P^u_{\widehat{G}},P^u_{G_x^*})\geq D(P^u_{\widehat{G}},P^u_{G_x^{**}})$ since penalty A does not hold for G_x^{**} and the average node degree of G_x^{**} greater than that of G_x^* by monotonicity of average node degree. Thus, the minimizer of the interior optimization must contain $\widehat{G}$ by the form of (B.6) (since $c(\cdot)>0$ and $\boldsymbol{q}$ is a probability vector). It is also clear that the minimizer must use the least possible number of edges to reach the class $\mathcal{G}_1$, as if more edges are used, we have their endpoints having larger KL divergences under OM2 between $\widehat{G}$ and minimizer than the minimizer with just enough edges removed the minimizer which are not in $\widehat{G}$.

   The minimum number of edges which must be added to $\widehat{G}$ to get into $\mathcal{G}_1$ is $\lceil\frac{N}{2}(\eta-\bar{d}_{\widehat{G}})\rceil$. Thus, since $\boldsymbol{x}_{E_{\widehat{G}}}=\mathbf{1}_{|E_{\widehat{G}}|}$, we only need to optimize

over the remaining components to get

$$\min_{q(u),u\in V} \min_{\boldsymbol{x}_{E_{\widehat{G}^C}}\in\{0,1\}^{|E_{\widehat{G}^C}|}:\boldsymbol{x}_{E_{\widehat{G}^C}} \text{ has } \lceil \frac{N}{2}(\eta-\bar{d}_{\widehat{G}})\rceil \text{ non-zero components}} \boldsymbol{q}\widetilde{\widetilde{M}}_{\widehat{G}^C}\boldsymbol{x}_{E_{\widehat{G}^C}} \tag{B.9}$$

2. $\widehat{G} \in \mathcal{G}_1$:

The proof for this case is relatively similar to the prior case. If the minimizer graph has edges outside $\widehat{G}$, then the KL divergences of the endpoints under OM2 increase by b_p times the number of times those endpoints occur (call this penalty B), and fewer edges of $\widehat{G}$ must be present in the minimizer to have a minimizer in $\mathcal{G}_0$. The KL divergences of the endpoints of the edges not in $\widehat{G}^C$ or the minimizer also increases by b_q times the number of edges with those endpoints. However, removing edges in the minimizer which are not in $\widehat{G}$ reduces the average node degree (giving a graph in $\mathcal{G}_0$). The KL divergences under OM2 are at least as low as the proposed minimizer (since penalty B is avoided). Thus, the objective function for fixed $\boldsymbol{q}, c(\cdot)$ has lower value when no edges in the minimizer exist outside $\widehat{G}$. Thus, the minimizer is contained in $\widehat{G}$. As in the prior case, it is easy to see that the minimizer will also contain the maximum number of edges in $\mathcal{G}_0$, as removing extra edges from $\widehat{G}$ increases the KL distances associated with their endpoints from $\widehat{G}$ to the minimizer under OM2. Thus, in (B.8) we see $\boldsymbol{x}_{E_{\widehat{G}^C}} = \mathbf{0}_{|E_{\widehat{G}^C}|}$ and $\lfloor \frac{N}{2}(\eta - \bar{d}_{\widetilde{G}})\rfloor$ edges must be removed from $\widehat{G}$ to form the minimizer.

Thus, the optimization problem in this case is

$$\min_{q(u),u\in V} \min_{\boldsymbol{x}_{E_{\widehat{G}^C}}\in\{0,1\}^{|E_{\widehat{G}}|}:\boldsymbol{x}_{E_{\widehat{G}}} \text{ has } \lfloor \frac{N}{2}(\eta-\bar{d}_{\widehat{G}})\rfloor \text{ non-zero components}} \boldsymbol{q}\widetilde{\widetilde{M}}_{\widehat{G}}(\mathbf{1}_{|E_{\widehat{G}}|}-\boldsymbol{x}_{E_{\widehat{G}}}) \tag{B.10}$$

which by noting $\mathbf{1}_{|E_{\widehat{G}}|} - \boldsymbol{x}_{E_{\widehat{G}}}$ inverts the entries of $\boldsymbol{x}_{E_{\widehat{G}}}$, we see that this is equivalent to

$$\min_{q(u),u\in V} \min_{\boldsymbol{x}_{E_{\widehat{G}^C}}\in\{0,1\}^{|E_{\widehat{G}}|}:\boldsymbol{x}_{E_{\widehat{G}}} \text{ has } \lfloor \frac{N}{2}(\eta-\bar{d}_{\widehat{G}})\rfloor \text{ zero components}} \boldsymbol{q}\widetilde{\widetilde{M}}_{\widehat{G}}\boldsymbol{x}_{E_{\widehat{G}}} \tag{B.11}$$

and by noting the length of $\boldsymbol{x}_{E_{\widehat{G}}}$, we see

$$\min_{q(u),u\in V} \min_{\boldsymbol{x}_{E_{\widehat{G}^C}}\in\{0,1\}^{|E_{\widehat{G}}|}:\boldsymbol{x}_{E_{\widehat{G}}} \text{ has } |E_{\widehat{G}}|-\lfloor\frac{N}{2}(\eta-\bar{d}_{\widehat{G}})\rfloor \text{ non-zero components}} \boldsymbol{q}\widetilde{\widetilde{M}}_{\widehat{G}}\boldsymbol{x}_{E_{\widehat{G}}} \tag{B.12}$$

to match the form of (B.9).

Note that the particular values of $-(p\alpha+(1-p)\beta)$ and $-(q\alpha+(1-q)\beta)$ are not important for the definition of (B.9) and (B.12) since they only affect terms which are subscripted with $\widehat{G}$ and $\widehat{G}^C$ and both do not appear in the same equation. Thus, we can replace them with their signs in the calculation of $\widetilde{\widetilde{M}}_{\widehat{G}}$ and $\widetilde{\widetilde{M}}_{\widehat{G}^C}$ to increase numerical stability.

The argument above is essentially equivalent to the game theory concept of *iterated elimination of dominated strategies* as described in Section 4.3 of [35]. The primary difference is the inner minimizations are equality constraints on binary vectors which typically have more than one entry which is 1. We can write this two-player zero sum game explicitly where the inner minimization would be over the standard basis of binary vectors by considering all subsets of the size of the minimizer (of which there are exponentially many), but this does not give us a useful computational form. It may be possible to further eliminate some graphs in the optimization problems (B.9) and (B.12), but in general, the result depends on the particular structure of the graph. There is a fundamental limit of simplifications which one can perform with the combinatorial structural constraints (which are encoded in $\widetilde{\widetilde{M}}$ through M).

Relaxing integrality constraints on the entries of $\boldsymbol{x}$ as in the case of OM1 gives us the control policies presented in Section 4.3.1.

REFERENCES

[1] Y. Xu, V. Olman, and D. Xu, "Clustering gene expression data using a graph-theoretic approach: An application of minimum spanning trees," *Bioinformatics*, vol. 18, no. 4, pp. 536–545, 2002.

[2] M. Newman, "The structure and function of complex networks," *SIAM Review*, vol. 45, no. 2, pp. 167–256, 2003.

[3] D. Easley and J. Kleinberg, *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge, UK: Cambridge University Press, 2010.

[4] M. Chiang, *Networked Life: 20 Questions and Answers*. Cambridge, UK: Cambridge University Press, 2012.

[5] Cooperative Association for Internet Data Analysis – University of California, San Diego, "IPv4 and IPv6 AS Core: Visualizing IPv4 and IPv6 Internet Topology at a Macroscopic Scale in 2013," Aug. 2013. [Online]. Available: http://www.caida.org/research/topology/as_core_network/

[6] J. Leskovec and C. Faloutsos, "Sampling from large graphs," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY: ACM, 2006, pp. 631 – 636.

[7] B. Ribeiro and D. Towsley, "Estimating and sampling graphs with multidimensional random walks," in *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*. New York, NY: ACM, 2010, pp. 390–403.

[8] J.-P. Onnela and N. A. Christakis, "Spreading paths in partially observed social networks," *Phys. Rev. E*, vol. 85, no. 3, p. 036106, Mar 2012.

[9] D. B. West, *Introduction to Graph Theory*, 2nd ed. Upper Saddle River, NJ: Prentice Hall, 2001.

[10] Oakland University, "Erdös number project," Aug. 2013. [Online]. Available: http://www.oakland.edu/enp/

[11] Wikipedia, "Erdös-Bacon number," Aug. 2013. [Online]. Available: http://en.wikipedia.org/wiki/Erdos-Bacon_number

[12] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, June 1998.

[13] C. Godsil and G. Royle, *Algebraic Graph Theory*. New York, NY: Springer, 2001.

[14] B. Huang and T. Jebara, "Maximum likelihood graph structure estimation with degree distributions," in *Analyzing Graphs: Theory and Applications, NIPS Workshop*, 2008.

[15] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

[16] H. Chernoff, "Sequential design of experiments," *Ann. Math. Statist.*, vol. 30, pp. 755–770, 1959.

[17] H. V. Poor, *An Introduction to Signal Detection and Estimation*, 2nd ed. New York, NY: Springer, 1994.

[18] B. Levy, *Principles of Signal Detection and Parameter Estimation*. New York, NY: Springer, 2008.

[19] S. M. Kay, *Fundamentals of Statistical Signal Processing, Volume 2: Detection Theory*. Upper Saddle River, NJ: Prentice-Hall, 1993.

[20] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. NY: John Wiley and Sons, Inc., 2006.

[21] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge University Press, 2004.

[22] B. Hajek, *Notes for ECE534: An Exploration of Random Processes for Engineers*. Urbana, IL: University of Illinois at Urbana-Champaign, 2011.

[23] L. Wasserman, *All of Statistics: A Concise Course in Statistical Inference*. New York, NY: Springer, 2010.

[24] A. Wald, *Sequential Analysis*. New York, NY: John Wiley and Sons, Inc., 1947.

[25] C. Baum and V. V. Veeravalli, "A sequential procedure for multiple hypothesis testing," *IEEE Trans. Inf. Theory*, vol. 40, pp. 1994–2007, 1994.

[26] R. M. Dudley, *Real Analysis and Probability*, 2nd ed. Cambridge, UK: Cambridge University Press, 2002.

[27] S. Nitinawarat, G. Atia, and V. Veeravalli, "Controlled sensing for multihypothesis testing," *IEEE Trans. Autom. Contr.*, vol. 58, no. 10, pp. 2451–2464, 2013.

[28] S. Nitinawarat, G. K. Atia, and V. V. Veeravalli, "Controlled sensing for hypothesis testing," in *Proceedings of the 37th International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Kyoto, Japan, Mar 2012, pp. 5277–5280.

[29] D. Koller and N. Megiddo, "The complexity of two-person zero-sum games in extensive form," *Games and Economic Behavior*, vol. 4, pp. 528–552, 1990.

[30] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer Networks and ISDN Systems*, vol. 30, no. 1, pp. 107–117, 1998.

[31] K. Avrachenkov, N. Litvak, M. Sokol, and D. Towsley, "Quick detection of nodes with large degrees," INRIA, Sophia Antipolis, France, Research Report 7881, Feb 2012.

[32] J. G. Ligo, G. K. Atia, and V. V. Veeravalli, "A controlled sensing approach to graph classification," in *Proceedings of the 38th International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vancouver, BC, Canada, May 2013, pp. 5573–5577.

[33] G. K. Atia and V. V. Veeravalli, "Controlled sensing for sequential multihypothesis testing," in *Proceedings of the 2012 IEEE International Symposium on Information Theory (ISIT)*, Cambridge, MA, July 2012, pp. 2196 – 2200.

[34] R. J. Vanderbei, *Linear Programming: Foundations and Extensions*, 2nd ed. New York, NY: Springer, 2001.

[35] M. J. Osborne and A. Rubinstein, *A Course in Game Theory.* Cambridge, MA: The MIT Press, 1994.